Diego Garcia Rodrigues

UM MODELO DE REDE NEURO-FUZZY BASEADA EM FUNÇÕES DE BASE RADIAL CAPAZ DE INFERIR REGRAS DO TIPO MAMDANI

Dissertação submetida ao Programa de Pós Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Ciência da Computação.
Orientador: Mauro Roisenberg, Dr.

Florianópolis

2015

Diego Garcia Rodrigues

UM MODELO DE REDE NEURO-FUZZY BASEADA EM FUNÇÕES DE BASE RADIAL CAPAZ DE INFERIR REGRAS DO TIPO MAMDANI

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre em Ciência da Computação", e aprovada em sua forma final pelo Programa de Pós Graduação em Ciência da Computação da Universidade Federal de Santa Catarina.

Florianópolis, 02 de fevereiro 2015.

Ronaldo dos Santos Mello, Dr.
Coordenador

Banca Examinadora:

Mauro Roisenberg, Dr.
Orientador

Rosa Maria Vicari, Dra.
Universidade Federal do Rio Grande do Sul

Silvia Modesto Nassar, Dra.
Universidade Federal de Santa Catarina

Paulo Jose de Freitas Filho, Dr.
Universidade Federal de Santa Catarina

Este trabalho é dedicado aos meus pais.

AGRADECIMENTOS

Agradeço aos meus pais, Carmen e Jamir, por todo afeto e dedicação, que foram cruciais para cada sucesso alcançado em minha vida. Muito obrigado, eu amo vocês.

Agradeço à minha madrinha, Carmem Vera, por ser uma pessoa que sempre esteve presente na minha vida, sendo praticamente uma segunda mãe para mim.

Agradeço à minha esposa, Viviane, por estar sempre ao meu lado me apoiando, até mesmo quando dificuldades surgem no caminho.

Agradeço ao meu orientador, prof. Mauro, pela sua presença constante no desenvolvimento do trabalho e por toda sabedoria e esforço que me foram dispensados. Mais uma vez, muito obrigado, professor.

Agradeço à banca examinadora, profª. Silvia, profª. Rosa e prof. Paulo, pelas contribuições para a melhoria do meu trabalho.

Agradeço aos meus amigos Luiz Angelo e Rafael Simon, pela amizade e por toda ajuda que me deram durante o desenvolvimento do meu trabalho.

Agradeço aos meus colegas de laboratório, em especial Mariana e Altieres, por todas conversas e por terem dividido uma parte importante de suas vidas comigo.

"Thus, the task is not so much to see what no one yet has seen, but to think what nobody yet has thought about that which everybody sees."

(Arthur Schopenhauer)

RESUMO

Este trabalho tem como objetivo apresentar um novo sistema de inferência neuro-fuzzy, chamado RBFuzzy, capaz de extrair conhecimento a partir de dados e gerar regras fuzzy do tipo Mamdani com alta interpretabilidade. A RBFuzzy é um sistema de inferência neuro-fuzzy que aproveita o comportamento funcional de neurônios ativados por Funções de Base Radial (RBF) e sua relação com sistemas de inferência fuzzy. A arquitetura da rede RBFuzzy permite extrair um conjunto de regras linguisticas a partir da estrutura conexionista e dos pesos ajustados de uma rede neural. Uma extensão do algoritmo da otimização da colônia de formigas (ACO, do inglês *ant colony optimization algorithm*) é utilizada para ajustar os pesos de cada regra para gerar um conjunto de regras fuzzy acurado e interpretável. Tendo um conjunto de regras fuzzy um especialista pode adicionar regras novas para incorporar conhecimento novo ao modelo de previsão gerado e também corrigir regras que foram geradas por dados imprecisos.

Palavras-chave: Fuzzy. Neuro-Fuzzy. Aprendizado de Máquina.

ABSTRACT

This work presents a novel neuro-fuzzy inference system, called RBFuzzy, capable of knowledge extraction and generation of highly interpretable Mamdani-type fuzzy rules. RBFuzzy is a four layer neuro-fuzzy inference system that takes advantage of the functional behavior of Radial Basis Function (RBF) neurons and their relationship with fuzzy inference systems. Inputs are combined in the RBF neurons to compound the antecedents of fuzzy rules. The fuzzy rules consequents are determined by the third layer neurons where each neuron represents a Mamdani-type fuzzy output variable in the form of a linguistic term. The last layer weights each fuzzy rule and generates the crisp output. An extension of the ant-colony optimization (ACO) algorithm is used to adjust the weights of each rule in order to generate an accurate and interpretable fuzzy rule set. For benchmarking purposes some experiments with classic datasets were carried out to compare our proposal with the EFuNN neuro-fuzzy model. The RBFuzzy was also applied in a real world oil well-log database to model and forecast the Rate of Penetration (ROP) of a drill bit for a given offshore well drilling section. The obtained results show that our model can reach the same level of accuracy with fewer rules when compared to the EFuNN, which facilitates understanding the operation of the system by a human expert.

Keywords: Fuzzy. Neuro-Fuzzy. Machine Learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

SUMÁRIO

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Técnicas de Aprendizado de Máquina (AM) tem como objetivo analisar um determinado problema e fornecer um modelo computacional que o represente (KOHAVI; PROVOST, 1998). Tom M. Mitchell define esse aprendizado como sendo uma questão de fazer com que os computadores consigam se programar com base em dados e em cima de uma estrutura inicial (MITCHELL, 2009). É esperado que o modelo resultante do aprendizado, além de fornecer a aproximação para os dados conhecidos, também forneça uma boa aproximação para o resultado de dados inéditos, que não foram apresentados ao modelo no aprendizado.

Uma das técnicas utilizadas no AM são as Redes Neurais Artificiais. Redes Neurais Artificiais são sistemas não lineares que utilizam um algoritmo específico de aprendizado. São aproximadores universais com uma boa capacidade de generalização e tolerância a ruído, porém o conhecimento do problema fica codificado nos pesos da rede neural, o que torna a interpretação do problema uma tarefa impraticável para um humano (CASTRO; MANTAS; BENÍTEZ, 2002). Por esse motivo Redes Neurais Artificiais são conhecidas como uma técnica "caixa-preta", i.e., determinar o motivo pelo qual uma Rede Neural Artificial fez uma determinada decisão é uma tarefa difícil (CASTRO; MANTAS; BENÍTEZ, 2002).

Sistemas de Inferência Fuzzy são sistemas baseados em um modelo linguístico para representação de conhecimento especialista. Esse modelo linguístico é composto de um conjunto de regras fuzzy do tipo SE-ENTÃO e conjuntos fuzzy (ZADEH, 1965). Assim como Redes Neurais Artificiais, sistemas de inferência fuzzy são aproximadores universais e podem ser utilizados para generalizar um problema (CASTRO; DELGADO, 1996). Sistemas de Inferência Fuzzy tem a vantagem de serem interpretáveis por utilizarem um modelo linguistíco para representar matematicamente o raciocínio humano. Um desafio dos Sistemas de Inferência Fuzzy é a construção direta do modelo a partir de dados conhecidos de um problema (BERTHOLD; HUBER, 1999).

Para solucionar essas limitações são utilizadas as Redes Neuro-Fuzzy. As Redes Neuro-Fuzzy combinam termos linguistícos e linguagem descritiva dos sistemas fuzzy com a capacidade de aprendizado das redes neurais artificiais. Desse modo os dados do problema são apresentados para a rede da mesma maneira que se faz com Redes Neurais Artificiais,

porém como resultado tem-se um Sistema de Inferência Fuzzy com toda sua capacidade descritiva. A partir deste modelo interpretável, um especialista pode entender o modelo e ajustá-lo, possibilitando contornar as limitações das técnicas de Aprendizado de Máquina citadas anteriormente (CASTRO; MANTAS; BENÍTEZ, 2002) (BERTHOLD; HUBER, 1999).

Entretanto essa abordagem apresenta um problema conhecido como Dilema da Interpretabilidade-Acurácia, no inglês "*Accuracy-Interpretability tradeoff*" (ZADEH, 1973).

Interpretabilidade se refere a representação do sistema do mundo real em uma maneira compreensível, enquanto acurácia se refere à precisão da resposta dada pelo modelo. Esses conceitos são contraditórios em Sistemas Fuzzy e depois de um certo limite se tornam quase mutuamente exclusivos (e.g., um modelo com apenas uma regra será altamente interpretável, mas não terá acurácia; um modelo com milhares de regras será altamente preciso, porém será incompreensível para um humano). Então, enquanto a complexidade do modelo aumenta, a qualidade da interpretabilidade diminui e a qualidade da precisão aumenta (veja a Figura 1). O problema é, então, conseguir o melhor balanço entre acurácia e interpretabilidade, para obter ao mesmo tempo um modelo preciso que seja o mais interpretável possível.

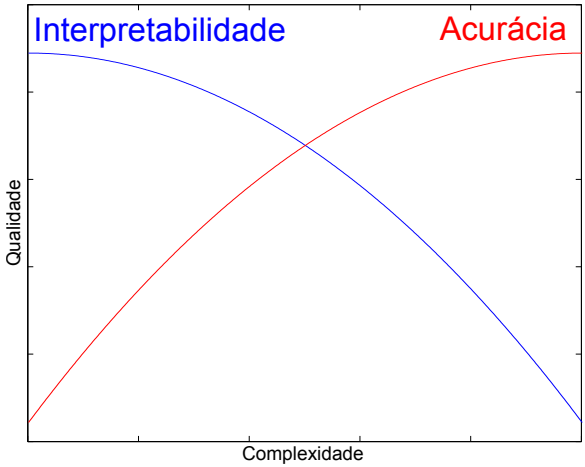

Figura 1: Dilema da Interpretabilidade-Acurácia
Fonte: Elaborada pelo autor

Na literatura são encontradas diversas propostas de sistemas

neuro-fuzzy, sendo que os modelos de rede neuro-fuzzy predominantes são a rede ANFIS, proposta em (JANG, 1993) e a família de redes ECoS proposta em (KASABOV, 1998). Uma das dificuldades encontradas no uso desses modelos é que os sistemas de inferência resultante são do tipo Takagi-Sugeno, que privilegia a acurácia em detrimento da interpretabilidade ((JASSBI et al., 2006), (JASSBI et al., 2007)). Há na família de redes ECoS um modelo específico de rede, chamado EFuNN, que gera como resultado um sistema de inferência do tipo Mamdani (MAMDANI, 1977), porém nos experimentos que foram conduzidos – e são mostrados no capítulo 4 – esse modelo teve um número excessivo de regras como resultado. Além disso, tanto na rede ANFIS quanto na família de redes ECoS, não é possível escolher qual será a importância dada para a interpretabilidade e acurácia, de modo que se possa favorecer modelos que sejam mais interpretáveis ou favorecer modelos que sejam mais acurados.

1.2 OBJETIVO GERAL

O objetivo deste trabalho é propor um novo modelo de rede neuro-fuzzy que seja o mais interpretável possível sem perder acurácia. O modelo proposto deve também possuir a capacidade de modificar a importância dada para os parâmetros interpretabilidade e acurácia de modo que a rede consiga se ajustar de acordo com a natureza do problema a ser enfrentado.

1.3 OBJETIVOS ESPECÍFICOS

Para atingir o objetivo proposto os seguintes objetivos específicos devem ser alcançados:

- Fazer uma revisão dos modelos de redes neuro-fuzzy existentes analisando vantagens e pontos fracos;
- Definir um modelo baseado na arquitetura de redes neurais RBF;
- Definir um algoritmo de ajuste dos pesos da rede neuro-fuzzy baseado na técnica de otimização por colônia de formigas;
- Definir uma função objetivo para este algoritmo de otimização que permita otimizar simultaneamente a acurácia e a interpretabilidade da rede;

- Propor e analisar experimentos, comparando com outras redes neuro-fuzzy.

1.4 ORGANIZAÇÃO DO TEXTO

Os capítulos desta dissertação estão organizados da seguinte maneira:

O Capítulo 2 procura fazer uma revisão dos conceitos básicos e trabalhos relacionados, com o objetivo de situar o leitor e criar uma visão geral do problema que está sendo tratado. Além disso, apresentar trabalhos já existentes na área e como eles tratam os problemas descritos.

O Capítulo 3 tem como objetivo mostrar o modelo proposto. Neste capítulo a arquitetura da rede proposta e o algoritmo de aprendizado são apresentados e exemplificados.

O Capítulo 4 apresenta os experimentos e resultados obtidos com o método proposto, utilizando dados simulados e reais, assim como a análise desses resultados e comparação com o resultado de outros modelos de rede neuro-fuzzy.

Ao final são apresentas as conclusões e sugestões para futuras pesquisas.

2 REVISÃO BIBLIOGRÁFICA

Neste capítulo serão apresentados os conceitos de sistemas de inferência fuzzy, interpretabilidade e acurácia, e redes neuro-fuzzy. Também serão apresentados os modelos de redes neuro-fuzzy predominantes na literatura.

2.1 ALGORITMOS DE COLÔNIAS DE FORMIGAS

Algoritmos de Colônias de Formigas (ACO) são algoritmos de otimização inspirados na observação do comportamento das formigas ao saírem de suas colônias para encontrar alimento. Essa técnica foi proposta em (DORIGO, 1992) e busca resolver problemas de otimização onde a solução se dá através da maximização ou minimização de uma função. Em geral o que se busca é a minimização do erro da função. Para atingir esse objetivo se faz necessário encontrar a melhor combinação de valores em diversas variáveis do problema.

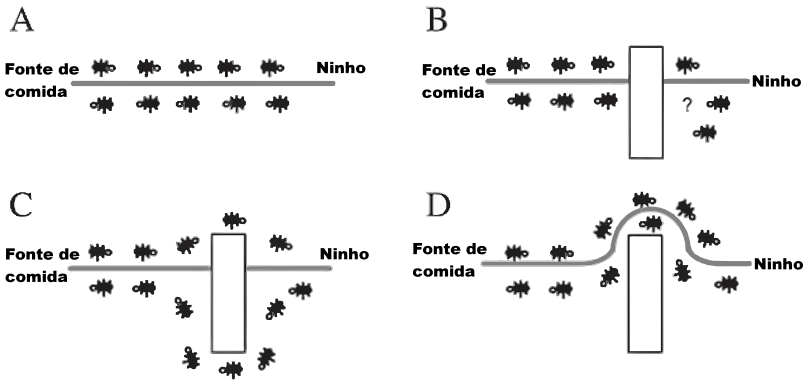

Figura 2: Comportamento das formigas durante a busca por alimento
Fonte: (CONTI; ROISENBERG; NETO, 2012)

Os algoritmos que se baseiam no ACO tentam explorar o seguinte comportamento das formigas: durante o processo de busca por alimentos, as formigas exploram de maneira aleatória a área próxima ao ninho e depositam nas trilhas por onde passam uma substância chamada

feromônio que evapora com o passar do tempo. As formigas tendem probabilisticamente a seguir a trilha de maior concentração de feromônio. A Figura 2 demonstra o comportamento na busca por alimento. Na Figura 2A as formigas seguem a trilha entre o alimento e o ninho. Na Figura 2B um obstáculo é inserido. Na Figura 2C, as formigas se dividem entre o caminho mais longo e o caminho mais curto. As formigas que foram pelo caminho mais curto farão mais viagens de ida e volta em menos tempo; fazendo com que mais feromônio se acumule nesse caminho, aumentando a probabilidade de que mais formigas sigam esse caminho. Isso faz com que o número de formigas no caminho mais longo diminuia e, consequentemente, faz com que o feromônio do caminho mais longo evapore. Isso diminui ainda mais a probabilidade de que uma formiga siga o caminho mais longo, como é exemplificado na Figura 2D (CONTI; ROISENBERG; NETO, 2012).

O comportamento descrito no parágrafo anterior faz com que o caminho menor (ótimo) prevaleça sobre o maior. Ao longo de mais iterações haverá um momento onde o menor caminho será significativamente mais atrativo e é nessa ideia que o ACO se baseia (Dorigo e Gambardella (apud CONTI; ROISENBERG; NETO, 2012)).

O algoritmo de otimização utilizado neste trabalho é o $ACO_{\mathbb{R}}V$ desenvolvido por Conti em (CONTI; ROISENBERG; NETO, 2012). Esse algoritmo estende a implementação do ACO para domínios contínuos e alcança os resultados com uma velocidade maior de convergência quando comparado com outras implementações (CONTI; ROISENBERG; NETO, 2012).

2.2 SISTEMAS DE INFERÊNCIA FUZZY

Sistemas de Inferência Fuzzy, também conhecidos como sistemas de inferência nebulosos ou sistemas de inferência difusos, são modelos computacionais utilizados com o objetivo de mapear uma determinada entrada para uma saída utilizando a teoria de Conjuntos Fuzzy, os conceitos de Lógica Fuzzy e conjuntos de regras do tipo SE-ENTÃO. A teoria de Conjuntos Fuzzy e os conceitos de Lógica Fuzzy fornecem uma base matemática sólida para que o modelo consiga representar a imprecisão de sistemas do mundo real (TANSCHEIT, 2004); como exemplo tem-se a representação de temperatura, que na lógica clássica seria representada de maneira binária como ‘quente‘ ou ‘frio‘, poderá ser representado com diferentes variações entre quente e frio, que é uma representação mais próxima de como a realidade funciona.

Nas próximas subseções a teoria de Conjuntos Fuzzy e os conceitos de Lógica Fuzzy serão explicados.

2.2.1 Conjuntos Fuzzy

A teoria de Conjuntos Fuzzy é uma extensão da teoria clássica dos conjuntos que foi formalizado por Lofti Zadeh em 1965 (ZADEH, 1965) com o objetivo de construir uma base matemática sólida para o tratamento de informações imprecisas ou aproximadas.

Na teoria clássica dos conjuntos tem-se como pertinência de um elemento a um determinado conjunto um valor verdadeiro ou falso. Isto é, dado um determinado conjunto A em um universo U pode-se representar a pertinência de elementos ao conjunto por uma função característica $f(x)$:

$$f(x) = \begin{cases} \mathbf{1} \text{ se e somente se x} \in \text{U} \\ \mathbf{0} \text{ se e somente se x} \notin \text{U} \end{cases}$$

que em resumo é uma definição binária, ou o elemento pertence totalmente ao conjunto ou não pertence.

Essa definição binária é estendida por Zadeh com a proposta de graus de pertinência mais abrangentes, que assumem valores intermediários no intervalo [0, 1]. Isto é, dado um determinado conjunto fuzzy B em um universo U pode-se representar a pertinência de elementos a este conjunto por uma função característica $\mu(x)_B$:

$$\mu(x)_B : U \rightarrow [0, 1],$$

onde $\mu(x)_B$ é uma função que irá indicar o grau de pertinência de x em B. Vários tipos diferentes de funções podem ser empregados como $\mu(x)_B$, os tipos mais comuns são funções trapezoidais, funções triangulares e, como no caso do tipo de função utilizado neste trabalho, funções de base radial. Para melhorar a interpretabilidade dos conjuntos fuzzy, o conceito de variável linguística é utilizado, que é uma variável cujos valores representam diferentes conjuntos fuzzy. (ZADEH, 1965)

2.2.1.1 Funções de Pertinência

As funções de pertinência são utilizadas para indicar o grau de compatibilidade de um elemento a um determinado conjunto fuzzy. Podem ser definidas por funções escolhidas de acordo com o problema e contexto onde serão utilizadas.

Alguns exemplos de funções geralmente utilizadas para representar funções de pertinência estão na figura 3.

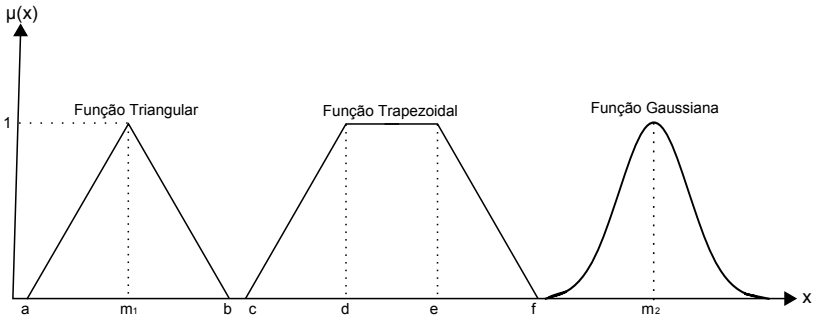

Figura 3: Exemplo de funções de pertinência
Fonte: Elaborada pelo autor

- Funções triangulares são definidas por um limite inferior a, um limite superior b e um valor m onde $a < m_1 < b$. São definidas pela equação 2.1:

$$trimf(x, a, m_1, b) = max\left(min\left(\frac{x-a}{m_1-a}, \frac{b-x}{b-m_1},\right), 0\right) \quad (2.1)$$

- Funções trapezoidais são definidas por um limite inferior c, um limite superior f, um limite de suporte inferior d e um limite de suporte superior e, onde $c < d < e < f$. São definidas pela equação 2.2:

$$trapmf(x, c, d, e, f) = max\left(min\left(\frac{x-c}{d-c}, 1, \frac{f-x}{f-e},\right), 0\right) \quad (2.2)$$

- Funções gaussianas são definidas por um valor central m_2 e um desvio padrão $k > 0$. Quando menor for k, mais estreita a função será. São definidas pela equação 2.3:

$$gaussmf(x, m_2, k) = e^{-\frac{1}{2}\left(\frac{x-m_2}{k}\right)^2} \quad (2.3)$$

2.2.1.2 Variáveis Linguísticas

Zadeh define variável linguística como sendo uma variável cujos valores são palavras ou sentenças em uma linguagem natural ou artificial (ZADEH, 1975). Como no exemplo dado anteriormente, a estatura pode ser considerada uma variável linguística assumindo os valores de baixa, média e alta. Esses valores são conjuntos fuzzy, representados por funções de pertinência, conforme o exemplo a seguir que utiliza funções trapezoidais e triangulares:

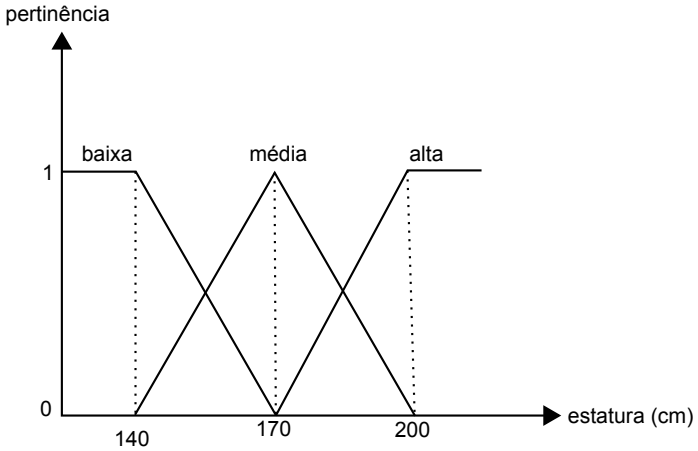

Figura 4: Funções de pertinência para conjuntos fuzzy associados à estatura

Fonte: Elaborada pelo autor

Essa variável linguística mapeia os valores de estatura numéricos para valores linguísticos, i.e., estatura baixa, estatura média e estatura alta. Para definir formalmente uma variável linguística é utilizada uma quíntupla $(N, T(N), X, G, M)$, onde N é o nome da variável; $T(N)$ é o conjunto de termos de N, isto é, a coleção dos valores linguísticos; X é o universo do conjunto; G é a regra sintática (gramática) para gerar os valores de N como uma composição de termos de $T(N)$, conectivos lógicos, modificadores e delimitadores; e M é a regra semântica (significado), para associar a cada valor gerado por G um conjunto fuzzy em X.

2.2.1.3 T-Normas e T-Conormas

T-Normas e T-Conormas são operações binárias que implementam os conectivos lógicos de conjunção e disjunção, respectivamente. Existem várias generalizações de T-Normas e T-Conormas na literatura (GEHRKE; WALKER; WALKER, 1999) (ZUO, 1995).

T-Norma (também conhecida como norma triangular) é a implementação da intersecção ou o operador lógico AND. Tem-se como definição da T-Norma:

Definição (T-Norma). *Uma t-norma é uma função* T*:* $[0,1]^2 \rightarrow [0,1]$ *que é: Simétrica, Associativa, Monotônica e 1 é o elemento neutro (ŁESKI, 2003) (TSOUKALAS; UHRIG, 1996).*

T-Conorma (também conhecida como conorma triangular) é a implementação da união ou o operador lógico OR. Tem-se como definição da T-Conorma:

Definição (T-Conorma). *Uma t-conorma é uma função* S*:* $[0,1]^2 \rightarrow [0,1]$ *que é: Simétrica, Associativa, Monotônica e 0 é o elemento neutro (ŁESKI, 2003) (TSOUKALAS; UHRIG, 1996).*

2.2.2 Lógica Fuzzy

Lógica Fuzzy, também conhecida como Lógica Difusa ou Lógica Nebulosa, é uma extensão da lógica clássica booleana que tem como objetivo suportar modos de raciocício aproximados. Enquanto na lógica clássica booleana são admitidos apenas dois valores lógicos possíveis: falso e verdadeiro , muitas vezes representados pelos números 0 e 1 respectivamente; na lógica fuzzy os valores lógicos podem possuir valores intermediários em sua representação que irão indicar o grau de veracidade das proposições. Na lógica fuzzy são utilizados conjuntos fuzzy e regras fuzzy para modelar o mundo e auxiliar a construção de modelos computacionais para tomada de decisão mais parecidas com decisões humanas (TANSCHEIT, 2004).

Ao utilizar conjuntos fuzzy para construir regras do tipo SE-ENTÃO, obtém-se regras fuzzy como no exemplo:

$$\text{Regra} = \begin{cases} \textbf{SE} \text{ temperatura } \textbf{É} \text{ fria ou temperatura } \textbf{É} \text{ muito fria} \\ \textbf{ENTÃO} \text{ ar condicionado deve aumentar temperatura} \end{cases}$$

2.2.3 Modelo Fuzzy Mamdani

O sistema de inferência fuzzy do tipo Mamdani foi proposto em (MAMDANI, 1974). Nesse sistema de inferência fuzzy as regras fuzzy são descritas de forma linguística, utilizando variáveis linguísticas que representam funções de pertinência. Um exemplo de regra fuzzy do tipo Mamdani é apresentado a seguir:

Se pressão é alta, então volume é pequeno.

onde alta e pequeno são variáveis linguísticas definidas por funções de pertinência.

2.2.4 Modelo Fuzzy Tsukamoto

Nos sistema de inferência fuzzy do tipo Tsukamoto, a parte consequente das regras fuzzy é dada por uma função de pertinência monotônica. Como resultado, cada regra terá um valor crisp dado pela força de ativação. A saída geral nesse sistema de inferência fuzzy será a média ponderada da saída de cada regra, o que acaba com a necessidade de um método de defuzzificação (TSUKAMOTO, 1979).

2.2.5 Modelo Fuzzy Sugeno

Nos sistemas de inferência fuzzy do tipo Sugeno, também conhecido como Takagi-Sugeno, a parte consequente das regras fuzzy é uma função crisp. Um exemplo de regra fuzzy do tipo Sugeno é dado a seguir:

SE x é A e y é B então z = f(x,y).

nesse tipo de sistema de inferência fuzzy a função *f(x,y)* é um polinômio que descreve a saída da regra para as entradas (SUGENO; TAKAGI, 1983).

2.2.6 Defuzzificação

É chamada de defuzzificação a etapa na qual a saída do sistema de inferência fuzzy é traduzida em um valor numérico(crisp). Existem vários tipos de defuzzificação propostos na literatura (LEEKWIJCK; KERRE, 1999), sendo os mais comuns:

- **Média dos Máximos:** o valor de saída é igual a média dos n elementos que correspondem aos maiores valores de pertinência do conjunto fuzzy de saída.
- **Média Ponderada dos Máximos:** o valor de saída é igual a média dos n elementos que correspondem aos maiores valores de pertinência do conjunto fuzzy de saída ponderado pelos graus de pertinência.
- **Centróide ou Centro de Gravidade:** o valor de saída é igual ao centro de gravidade do conjunto de saída do sistema fuzzy.
- **Critério Máximo ou Mínimo:** o valor de saída é igual ao valor máximo (ou mínimo) de ativação do conjunto de saída do sistema fuzzy.

2.3 INTERPRETABILIDADE E ACURÁCIA

Acurácia em um modelo de previsão pode ser definido como a capacidade do modelo representar de maneira precisa um sistema do mundo real (CASILLAS et al., 2003). Essa capacidade pode ser medida como um percentual de padrões classificados de maneira correta em um modelo de classificação ou como o erro médio quadrático(MSE) em um modelo de regressão. Geralmente o foco principal ao modelar um sistema de previsão é obter o máximo de acurácia possível.

Interpretabilidade é considerada uma das maiores vantagens de sistemas de inferência fuzzy sobre modelos de previsão tradicionais (MIKUT; JÄKEL; GRÖLL, 2005), como as redes neurais artificiais.

Interpretabilidade é a habilidade de um modelo de representar um sistema do mundo real em uma maneira compreensível (CASILLAS et al., 2003). Quanto mais compreensível para um leitor humano o modelo é, mais interpretável ele é considerado. Interpretabilidade não é tão fácil de ser medida quanto acurácia, e geralmente há controvérsia quanto a escolha do método ótimo para medi-la ((GUILLAUME, 2001), (ZHOU; GAN, 2008), (ALONSO; MAGDALENA; GONZÁLEZ-RODRÍGUEZ,

2009), (GACTO; ALCALÁ; HERRERA, 2011)). A medida utilizada nesta dissertação para avaliar a interpretabilidade de um modelo é baseada no número de regras do modelo. Com relação ao número de regras, um modelo ótimo seria considerado o modelo com o menor número de regras possível.

Quando sistemas de inferência fuzzy são construídos a partir de conhecimento de um especialista, geralmente o modelo apresenta um nível de interpretabilidade alto e uma acurácia satisfatória. Porém quando um sistema de inferência fuzzy é extraído de dados, a maioria dos métodos na literatura foca apenas em otimizar a acurácia (GUILLAUME; MAGDALENA, 2006).

2.4 REDES NEURO-FUZZY

O termo Redes Neuro-Fuzzy se refere à combinação de Redes Neurais Artificiais com Sistemas de Inferência Fuzzy. Nesses sistemas são incorporadas as técnicas de aprendizado automático de redes neurais artificiais com a capacidade de representação próxima do raciocínio humano oferecido por sistemas Fuzzy. Ou seja, há uma fase de treinamento supervisionado, no qual são apresentados os dados em pares entrada/saída e como resultado há um modelo linguístico da representação do conhecimento extraído dos dados na forma de um sistema de inferência fuzzy (JANG; SUN, 1995).

Devido a esta natureza híbrida, as redes neuro-fuzzy herdam as características de redes neurais e sistemas de inferência fuzzy, ou seja, além do aprendizado automático a partir de dados que fornece conhecimento implícito, as regras podem ser inseridas no sistema de inferência a partir do conhecimento explícito fornecido por um especialista. As regras geradas a partir dos dados também podem ser otimizadas utilizando o conhecimento do especialista (JANG; SUN, 1995).

O tipo de aprendizado desse tipo de rede pode ser dividido entre aprendizado *Online* e aprendizado *Offline*. No aprendizado *Online* a rede vai se modificando dinamicamente enquanto os dados são apresentados, de modo que ao fornecer mais dados para a rede ela não precisa ser treinada novamente com os dados previamente apresentados. No aprendizado *Offline* a rede só é modificada após a apresentação de todos os dados do conjunto de treinamento e, caso novos dados tenham que ser apresentados, todos o conjunto de treinamento deve ser apresentado novamente para a rede (KASABOV; SONG, 2002).

Uma das principais motivações para o uso de redes neuro fuzzy

é fornecer um mecanismo formal para transformar experiência (dados) e conhecimento de especialistas humanos em uma representação do conhecimento de maneira explícita na forma de um sistema de inferência fuzzy (JANG, 1993).

Nas próximas subseções são descritas as redes neuro-fuzzy predominantes na literatura, a rede ANFIS proposta por Jang e a família de redes ECoS proposta por Kasabov.

2.4.1 ANFIS

O Sistema de Inferência Neuro-Fuzzy Adaptativo, no inglês *"Adaptive Neural Fuzzy Inference System"* (ANFIS), foi proposto por Jang. O ANFIS é uma Rede Neural Generalizada, no inglês *"Generalized Neural Network"* (GNN), composta por uma rede feed-forward com um procedimento de aprendizado de descida de gradiente, que é utilizada para construir um Sistema de Inferência Fuzzy do tipo Takagi-Sugeno (JANG, 1993).

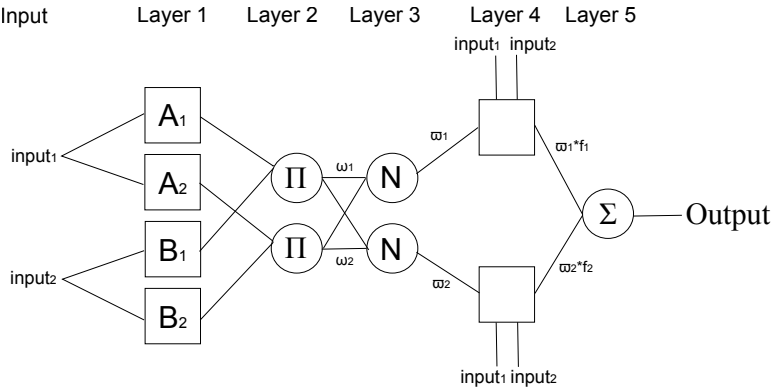

Figura 5: Arquitetura da rede Anfis
Fonte: (JANG, 1993)

ANFIS é composto de cinco camadas de neurônios, como exibido na Figura 5. Cada uma dessas camadas possui diferentes tipos de neurônios, alguns são descritos por uma função específica. Há dois tipos básicos de neurônios: os adaptativos e os fixos. Os neurônios adaptativos podem ser adicionados e modificados, enquanto os fixos podem apenas ter modificações nos pesos de suas conexões neurais.

A primeira camada é composta de neurônios adaptativos, cada

um representa uma função de pertinência fuzzy para cada entrada. A segunda camada é composta de neurônios com a função "produto", sua saída é o produto de todos os sinais de entrada.

A terceira camada é a camada de normalização, sua saída é a soma de todos os sinais de entrada ponderados por ϖ_i que em seguida é normalizado.

A quarta camada representa a parte consequente das regras, onde cada neurônio é do tipo adaptativo, sua saída é o resultado de uma função que leva como parâmetro o sinal de entrada da terceira camada e as entradas da rede.

A quinta camada faz o cálculo da soma de todos os sinais de entrada ponderados por $\varpi_i * f_i$.

Apesar de ser a primeira proposta de rede neuro-fuzzy encontrada na literatura, há pesquisas recentes que atribuem bons resultados para ANFIS ((OCAK; ERTUNC, 2012), (CHANG; WANG, 2013)). A grande desvantagem da ANFIS é o fato de que ela utiliza um sistema de inferência fuzzy do tipo Takagi-Sugeno ((JANG, 1993)). Takagi-Sugeno possui uma interpretabilidade mais baixa quando comparado com modelos do tipo Mamdani ((JASSBI et al., 2006), (JASSBI et al., 2007)).

2.4.2 ECoS

Sistemas conexionistas evolutivos, no inglês *Evolving Connectionist Systems* (ECoS), são uma família de redes neurais artificiais proposta por Kasabov em (KASABOV, 1998). Evolutivos no contexto do ECoS não está relacionado a evolutivo no contexto de algoritmos genéticos. Evolutivo para ECoS se refere ao fato de que esses sistemas mudam ao longo do tempo, a estrutura da rede do ECoS é dinâmica. A Figura 6 mostra a arquitetura básica de um sistema ECoS. A maior característica do ECoS é o algoritmo de aprendizagem que modifica a estrutura da rede enquanto os exemplos de treinamento(dados) são apresentados para a rede. Todos os sistemas dessa família seguem 3 princípios básicos (KASABOV, 1998):

- Algoritmo de aprendizado de uma passada. Isto é, na primeira passagem do exemplo de treinamento para a rede, ela já deve aprender o dado.

- O aprendizado é "online". Isto é, enquanto dados são apresentados para a rede ECoS, os dados que foram apresentados nos treinamentos passados não precisam ser reapresentados e não são

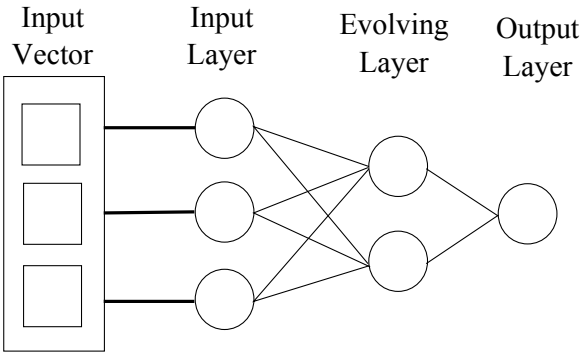

Figura 6: Arquitetura básica da família Ecos
Fonte: (WATTS, 2009)

"esquecidos". A arquitetura geral de um sistema ECoS e do algoritmo de aprendizado possibilitam que uma rede ECoS acomode novos dados sem perder a capacidade adquirida com os dados apresentados em treinamentos anteriores.

- Os neurônios são adicionados quando dados são apresentados. Inicialmente alguns dados são armazenados, porém, quando novos dados são apresentados para a rede, os neurônios existentes podem ser modificados ou novos neurônios podem ser adicionados. Caso um neurônio adicionado não seja modificado é possível extrair qual dado foi responsável pela sua origem.

2.4.2.1 DENFIS

O Sistema de inferência neuro-fuzzy dinâmico, no inglês *Dynamic evolving neural-fuzzy inference system* (DENFIS), é uma aplicação especial da família ECoS, proposto por Kasabov em (KASABOV; SONG, 2002). Há dois tipos de DENFIS, um tipo usa o algoritmo de aprendizado *Online* e o outro utiliza um algoritmo de aprendizado *Offline* para casos onde há possibilidade do uso de um aprendizado *Offline* para o objetivo de otimização. Ambos os tipos utilizam o sistema de inferência fuzzy do tipo Takagi-Sugeno. Aprendizado do tipo *Online* é uma boa alternativa para sistemas onde dados são continuamente gerados e apresentados para o sitema. Aprendizado *Offline* é, na maioria dos casos, mais rápido e tende a ter um erro de previsão médio menor que o aprendizado online

(KASABOV, 2007).

O mecanismo básico utilizado pelo DENFIS para gerar regras é um algoritmo de clusterização chamado Método de Clusterização Evolutivo, do inglês *Evolving Clustering Method* (ECM). Os agrupamentos resultantes(no inglês *clusters*) são utilizados para particionar o espaço de entrada dos dados, depois uma rede neural artificial do tipo MLP, com o algoritmo backpropagation, é utilizada para achar os valores ótimos para a parte consequente das regras fuzzy. A parte antecedente das regras fuzzy é determinada pelos centros dos agrupamentos.

As regras criadas pela DENFIS são definidas na forma:

$$\begin{cases} \textbf{se } \mathrm{x_1} \text{ é } \mathrm{R_{11}} \text{ e } \mathrm{x_2} \text{ é } \mathrm{R_{12}} \text{ e } \ldots \mathrm{x_q} \text{ é } \mathrm{R_{1q}}, \\ \textbf{então } \mathrm{y} \text{ é } \mathrm{f_1(x_1,x_2,\ldots,x_q)} \\ \textbf{se } \mathrm{x_1} \text{ é } \mathrm{R_{21}} \text{ e } \mathrm{x_2} \text{ é } \mathrm{R_{22}} \text{ e } \ldots \mathrm{x_q} \text{ é } \mathrm{R_{2q}}, \\ \textbf{então } \mathrm{y} \text{ é } \mathrm{f_2(x_1,x_2,\ldots,x_q)} \\ \textbf{se } \mathrm{x_1} \text{ é } \mathrm{R_{m1}} \text{ e } \mathrm{x_2} \text{ é } \mathrm{R_{m2}} \text{ e } \ldots \mathrm{x_q} \text{ é } \mathrm{R_{mq}}, \\ \textbf{então } \mathrm{y} \text{ é } \mathrm{f_m(x_1,x_2,\ldots,x_q)} \end{cases}$$

sendo m o número de centros dos agrupamentos e q o tamanho do vetor de entrada. $\boldsymbol{x_j}$ ***é*** $\boldsymbol{R_{ij}}$ são os $\boldsymbol{m}$ $\boldsymbol{x}$ $\boldsymbol{q}$ antecedentes para as $\mathbf{m}$ regras. $\mathbf{x_j}$ são as variáveis antecedentes e $\mathbf{R_{ij}}$ são os conjuntos fuzzy. A parte consequente das regras, y é a variável consequente e funções $\mathbf{f_m}$ são empregadas.

Para um vetor de entrada $\mathbf{x^k = [x_1^k, x_2^k, \ldots, x_q^k]}$ o resultado da inferência y^k é definido pela equação 2.4:

$$\mathrm{y}^k = \frac{\Sigma_{i=1}^{m} = \omega_i f_i(\mathrm{x}_1^k, \mathrm{x}_2^k, \ldots, \mathrm{x}_q^k)}{\Sigma_{i=1}^{m} = \omega_i}, \quad \text{onde } \omega_i = \Pi_{j=1}^{q} \omega_{Rij}(xj^k) \tag{2.4}$$

Para achar a função $\mathbf{f_i}$ são utilizadas redes neurais do tipo MLP .

A Figura 7 mostra o resumo do algoritmo de aprendizagem. Isso sumariza o modelo DENFIS. Uma explicação mais aprofundada pode ser encontrada em (KASABOV; SONG, 2002).

2.4.2.2 EFuNN

Redes Neurais Fuzzy Evolutivas, no inglês *Evolving Fuzzy Neural Networks* (EFuNN), é outra aplicação especial de ECoS e foi proposta

1: Use o algoritmo ECM para clusterizar os dados do vetor de entrada
2: Otimizar os clusters (passo opcional no algoritmo)
3: Clusterizar os dados de treinamento em N_RULES clusters.
4: Criar a parte antecedente para cada regra fuzzy
5: Estimar a função **f** para cada regra fuzzy, para isso treinando uma rede neural do tipo MLP com os dados correspondentes.

Figura 7: Overview do algoritmo de aprendizagem do DENFIS

em (KASABOV, 1998). É basicamente uma rede neural *feed-forward* de cinco camadas. A primeira camada de neurônios é a camada de entrada. A segunda camada é a camada de condições. Cada neurônio na camada de condições representa uma função de pertinência fuzzy para uma entrada. A terceira camada de neurônios é a camada evolutiva, onde regras fuzzy são adicionadas e armazenadas. A quarta camada é a camada de ação, os neurônios dessa camada representam funções de pertinência fuzzy ligadas aos neurônios de saída. A quinta camada é a camada de saída, onde o valor crisp é calculado da saída fuzzy da quarta camada utilizando um método de defuzzificação.

Regras fuzzy do tipo Mamdani podem ser extraidas de uma EFuNN já treinada utilizando um algoritmo específico, descrito em (KASABOV; WOODFORD, 1999).

Bons resultados mostrando rápida performance e baixo erro já foram atribuídas para a rede EFuNN na literatura (CHANG; FAN; LIN, 2011), (ALMOMANI et al., 2012). Entretanto as desvantagens atribuídas para a EFuNN são o fato de que algumas variáveis, como número e tipo de funções de pertinência para cada variável de saída, são calculadas automaticamente e não podem ser modificadas e os parâmetros ótimos para o aprendizado são difíceis de serem configurados (PETROVIC-LAZAREVIC; ZHANG, 2009).

3 PROPOSTA

Para conseguir alcançar um modelo com um nível alto de acurácia e ao mesmo tempo um alto nível de interpretabilidade, um modelo com uma abordagem diferente das redes neurais apresentadas deve ser utilizado. O modelo proposto nesta dissertação, chamado de RBFuzzy, utiliza um algoritmo de otimização com uma função de aptidão(no inglês *fitness*) desenvolvida para satisfazer essas duas condições, o qual vai ser descrito nas próximas seções.

3.1 RBFUZZY

O modelo proposto, chamado de RBFuzzy, é uma rede neuro-fuzzy que implementa regras fuzzy do tipo Mamdani. Similarmente ao algoritmo proposto em (LI; HORI, 2006), que dá uma interpretação fuzzy a redes neurais RBF, a RBFuzzy também usa neurônios ativados por funções de base radial e um algoritmo de clusterização. Porém, diferentemente da proposta de (LI; HORI, 2006), a RBFuzzy usa neurônios que representam termos linguisticos para a variável de saída.

A RBFuzzy pode ser treinada usando um algoritmo híbrido de aprendizado, o que ocorre em dois passos. No primeiro passo, um algoritmo de clusterização determina as funções de pertinência para as variáveis de entrada baseado na distribuição dos dados no espaço de entrada. Num segundo passo um algoritmo de otimização, nesse caso o $ACO_{\mathbb{R}}V$ desenvolvido por Conti em (CONTI; ROISENBERG; NETO, 2012), determina os pesos da rede. Através do uso do $ACO_{\mathbb{R}}V$ é possível incorporar mais de um objetivo no algoritmo de aprendizado, otimizando dessa forma a acurácia e a interpretabilidade ao mesmo tempo.

As próximas subseções são organizadas da seguinte maneira. A subseção 3.1.1 descreve a arquitetura da RBFuzzy. A subseção 3.1.2 descreve como o algoritmo de aprendizagem funciona. A subseção 3.1.3 explica como converter a rede treinada na sua forma linguistica de regras. Finalmente, a subseção 3.1.4 apresenta as vantagens e desvantagens da RBFuzzy.

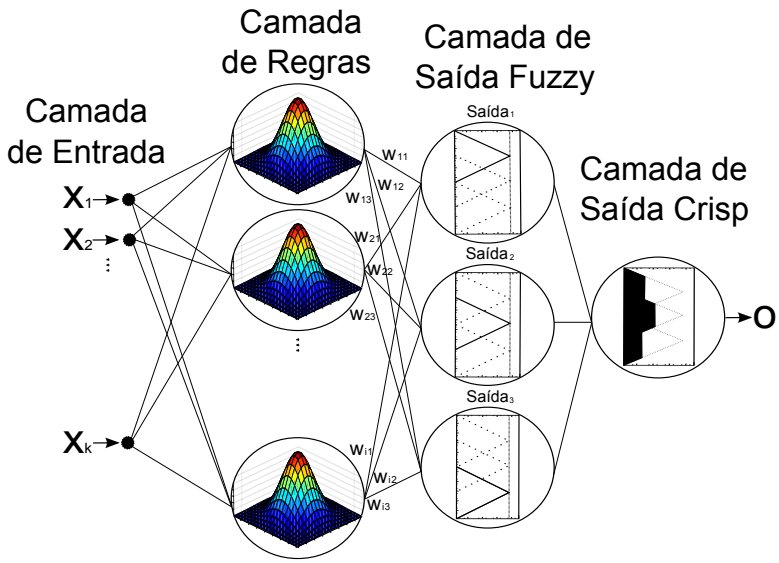

Figura 8: Arquitetura RBFuzzy
Fonte: Elaborada pelo autor

3.1.1 Arquitetura

A arquitetura está apresentada na Figura 8. É basicamente uma rede neural de 4 camadas. Essa arquitetura de rede pode ser interpretada como um Sistema de Inferência Fuzzy, o que vai ser explicado nas seções a seguir.

A primeira camada é a camada de entrada, onde os dados são apresentados para a rede e que representa as variáveis de entrada.

A segunda camada é chamada de Camada de Regras. Cada neurônio da Camada de Regras representa um conjunto de funções de pertinência fuzzy e é gerado como resultado de um algoritmo de clusterização nos conjunto de dados de treinamento. Ao usar um algoritmo de clusterização, cada neurônio irá representar um agrupamento no espaço de entrada. A distribuição dos dados de cada agrupamento é utilizado para definir os parâmetros da função de ativação para cada variável de entrada. A função de ativação da Camada de Regras é uma Função de Base Radial (RBF). Para cada variável de entrada, uma função Gausiana é calculada e a ativação de um dado neurônio i é calculado como sendo o mínimo de todas as gaussianas de entrada,

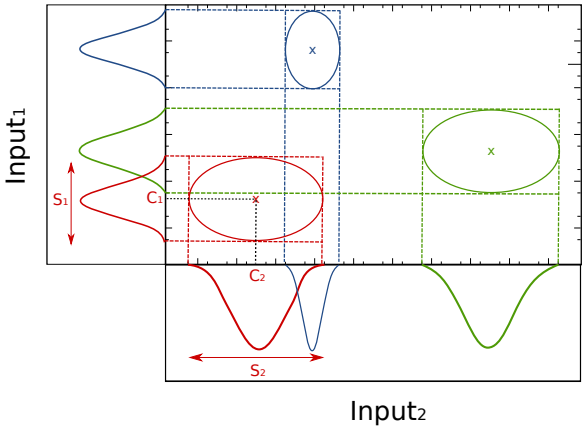

Figura 9: Antecedentes - Camada de Regras
Fonte: Elaborada pelo autor

como mostrado na equação 3.1 pela função g_i:

$$f(x, c, s) = exp\left(-\frac{(x-c)^2}{2 * s^2}\right)$$
$$g_i = min(f(x_1, c_1, s_1), \ldots, f(x_k, c_k, s_k)) \tag{3.1}$$

onde x_k representa uma variável de entrada, c_k e s_k representam o centro e valor de *spread* da função de pertinência respectivamente. Para cada agrupamento, c_k é definido como sendo igual à média dos dados do agrupamento e s_k como sendo o desvio padrão dos dados do agrupamento.

O número de neurônios da Camada de Regras é um parâmetro escolhido pelo usuário para controlar o número de agrupamentos no espaço de entrada. Um número maior de neurônios irá resultar em um número maior de regras, uma vez que o número de neurônios irá determinar o número de funções de pertinência para cada variável de entrada. Em cada neurônio da Camada de Regras as variáveis de entrada são combinadas para formar os antecedentes das regras fuzzy como mostrado na Figura 9. Por exemplo, vamos supor que um dado neurônio da Camada de Regras possui duas variáveis de entrada com os parâmetros (c_1, s_1) para Input_1 e (c_2, s_2) para Input_2 como pode ser visto no agrupamento vermelho da Figura 9. Neste caso o antecedente das regras fuzzy representadas por este neurônio específico será: *"SE Input_1 ESTÁ EM gaussmf(c1, s1) E Input_2 ESTÁ EM gaussmf(c2, s2)*

ENTÃO...". O consequente é definido na camada de Saída Fuzzy.

Quando um vetor de entrada é submetido para a rede, a ativação de saída dos neurônios da Camada de Regras descreve o grau de pertinência deste vetor de entrada para cada regra representada pelos neurônios. As conexões w_{ij} da Camada de Regras para a Camada de Saída Fuzzy representam o peso das regras para cada saída fuzzy, estes pesos são ajustados pelo algoritmo $\text{ACO}_{\mathbb{R}}\text{V}$. Esse passo é explicado em detalhes na seção 3.1.2.

A terceira camada é a Camada de Saída Fuzzy. O número de neurônios nessa camada é um parâmetro escolhido pelo usuário e deve ser igual ao número de conjuntos fuzzy de saída desejados, e.g., ao definir três como o número de neurônios nessa camada, três conjuntos de saída serão gerados e serão linguísticamente interpretados como: BAIXO, MÉDIO E ALTO. As funções empregadas nas funções de pertinência dos conjuntos de saída são do tipo Triangular.

A saída de cada neurônio j na Camada de Saída Fuzzy representa o grau de pertinência do vetor de entrada para a variável de saída fuzzy, representado pela função g_i, ponderada pelo peso w_{ij}. Como resultado tem-se a seguinte função de ativação para a Camada de Saída fuzzy, definida pela equação 3.2:

$$z_j = max(w_{1j} * g_1, w_{2j} * g_2, \ldots, w_{ij} * g_i), \tag{3.2}$$

Cada neurônio da Camada de Regras possui conexão para todos os neurônios da Camada de Saída Fuzzy. Dessa maneira, o número total de regras geradas pelo modelo RBFuzzy é $i \times j$, onde i é o número total de neurônios na Camada de Regras e j é o número de neurônios na camada de Saída Fuzzy. Note que para cada neurônio da Camada de Regras, j regras são criadas, cada uma com o mesmo antecedente e com um consequente diferente. Cada uma dessas regras terá um peso diferente que será determinado pelo algoritmo de aprendizado. Note também que regras com pesos pequenos serão cortadas do conjunto final de regras para melhorar a interpretabilidade do modelo, pois não afetam significativamente o modelo. O exemplo a seguir mostra as regras geradas quando o neurônio da Camada de Regras representado pela região vermelha da Figura 9 é conectado aos neurônios da Camada de Saída Fuzzy:

$$\text{Regra 1} = \begin{cases} \textbf{SE } \text{Input}_1 \text{ É gaussmf}(\text{c}_1, \text{ s}_1) \text{ E} \\ \textbf{SE } \text{Input}_2 \text{ É gaussmf}(\text{c}_2, \text{ s}_2) \\ \textbf{ENTÃO } \text{SAÍDA É ALTA COM w} = (\text{w}_{11}) \end{cases}$$

$$\text{Regra 2} = \begin{cases} \textbf{SE } \text{Input}_1 \text{ É gaussmf}(c_1, s_1) \text{ E} \\ \textbf{SE } \text{Input}_2 \text{ É gaussmf}(c_2, s_2) \\ \textbf{ENTÃO SAÍDA É MÉDIA COM w} = (w_{12}) \end{cases}$$

$$\text{Regra 3} = \begin{cases} \textbf{SE } \text{Input}_1 \text{ É gaussmf}(c_1, s_1) \text{ E} \\ \textbf{SE } \text{Input}_2 \text{ É gaussmf}(c_2, s_2) \\ \textbf{ENTÃO SAÍDA É BAIXA COM w} = (w_{13}) \end{cases}$$

A quarta e última camada é a Camada de Saída Crisp, que computa o valor numérico da saída da rede. Neste trabalho, este valor é computado através da aplicação direta do método de defuzzificação do centróide nas saídas da Camada de Saída Fuzzy, porém outros métodos de defuzzificação podem ser aplicados.

3.1.2 Algoritmo de Aprendizagem

A Figura 10 mostra o processo básico do algoritmo de aprendizagem que a RBFuzzy implementa. No primeiro passo o número de neurônios RBF da Camada de Regras deve ser escolhido pelo usuário. O algoritmo poderia também escolher o número ótimo de neurônios RBF utilizando algum tipo de algoritmo de clusterização especializado, como por exemplo o algoritmo de clusterização ECM (KASABOV; SONG, 2002).

No segundo passo, o número de funções de saída fuzzy são escolhidos. Recomenda-se o uso de valores entre 3 e 5 funções de pertinência de saída. Em seguida os dados são divididos utilizando um algoritmo de clusterização. O algoritmo de clusterização utilizado em todos experimentos desse trabalho é o algoritmo k-means++ (ARTHUR; VASSILVITSKII, 2007). Depois os dados são analisados estatisticamente para definir os parâmetros das funções de base radial, nesse passo os centros e valores de *spread* para cada função de pertinência de entrada são definidas para cada agrupamento separadamente, a Figura 11 mostra o resultado final desse passo. No passo final o $\text{ACO}_{\mathbb{R}}\text{V}$ é utilizado para minimizar o erro e minimizar o peso total das regras empregando uma função de aptidão apropriada, que é mostrada na equação 3.3. Os pesos das regras são minimizados porque regras com pesos muito pequenos fazem com que não exista muita influência na saída da rede e são descartadas no passo de Poda de Regras, fazendo com que o modelo fique ainda mais interpretável. Esse processo é explicado de maneira

mais aprofundada a seguir.

1: $N\_REGRAS \leftarrow$ Escolha o número de regras
2: $N\_SAIDA \leftarrow$ Escolha o número de funções de pertinência de saída
3: $CLUSTERS \leftarrow$ Use um algoritmo de clusterização para dividir o conjunto de treinamento em N_REGRAS agrupamentos
4: **for all** cluster$_i$ in CLUSTERS **do**
5: Defina o centro da função de base radial da regra$_i$ como sendo igual ao centro do cluster$_i$.
6: Defina o valor de *spread* da função de base radial da regra$_i$ como sendo igual ao desvio padrão do cluster$_i$
7: **end for**
8: Use o algoritmo ACO$_{\mathbb{R}}$V para achar os pesos ótimos para a rede
9: Descarte as regras que estão com os pesos abaixo de um limite determinado pelo usuário

Figura 10: Funcionamento Básico do Algoritmo de Aprendizagem da rede RBFuzzy
Fonte: Elaborada pelo autor

O algoritmo de aprendizagem pode ser dividido em duas fases, a primeira sendo responsável pela definição do espaço de entrada coberto por cada regra, ilustrado na figura 10 pelas linhas 1 até 7, e a segunda fase sendo responsável por definir os pesos ótimos da rede e descarte de regras desnecessárias, ilustrado na figura 10 pelas linhas 8 e 9.

Na primeira fase os dados de treinamento são divididos em i regiões no espaço de entrada, sendo i igual ao número de neurônios RBF na Camada de Regras. Para atingir esse objetivo, um algoritmo de clusterização é utilizado para particionar o conjunto de dados de treinamento em i agrupamentos.

A função de ativação dos neurônios da camada de Regras é definido pela equação 3.1. A Figura 11 mostra o resultado desse processo, onde cada agrupamento está reprensentado por uma cor diferente.

Na segunda fase o algoritmo ACO$_{\mathbb{R}}$V (CONTI; ROISENBERG; NETO, 2012) é utilizado para encontrar os pesos entre a camada de Regras e a Camada De Saída Fuzzy. O objetivo dessa fase é minimizar o erro da rede e minimizar os pesos da rede simultâneamente. A função de aptidão utilizada pelo algoritmo ACO$_{\mathbb{R}}$V é definida pela equação 3.3

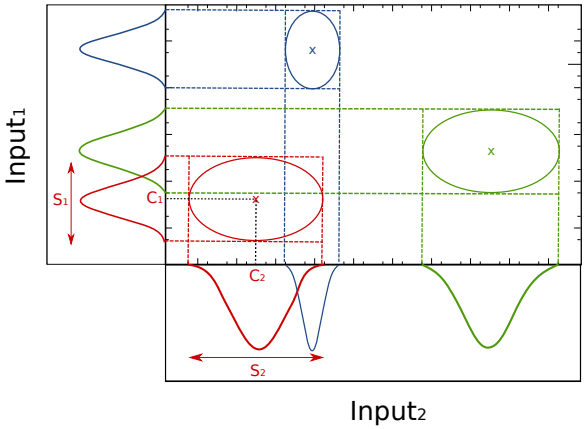

Figura 11: Funções de Pertinência de Entrada - Camada de Regras
Fonte: Elaborada pelo autor

$$fit = \alpha\sqrt{\frac{1}{N}\sum_{i=1}^{N}(o_i - t_i)^2} + \lambda\sum_{i=1}^{m}\sum_{j=1}^{n}(w_{ij}) \tag{3.3}$$

onde α e λ são respectivamente os pesos para acurácia e interpretabilidade, N é o número total de pontos de treinamento contidos no conjunto de dados de treinamento, o_i é o valor de saída para o dado de treinamento i, t_i é o valor esperado para a saída do dado de treinamento i, e w_{ij} é o peso entre o neurônio i da Camada de Regras para o neurônio j da Camada de Saída Fuzzy. Os valores de α e λ são determinados através de experimentação; quando o erro da rede deve ser diminuido deve-se aumentar o valor de α e quando se deseja aumentar a interpretabilidade o valor de λ deve ser aumentado.

A função de complexidade deve ser modelada baseada na medida de interpretabilidade escolhida, pois conforme apresentado na seção 2.3 existem várias medidas (definições) de interpretabilidade. Neste trabalho a medida utilizada para avaliar a interpretabilidade é o número de regras, dessa maneira a função de complexidade é a soma ponderada de todos os pesos de todas as regras do modelo. Adotando esses parâmetros o algoritmo ACO$_{\mathbb{R}}$V irá favorecer modelos com pesos menores. Uma vez que regras com pesos baixos quase não possuem influência na saída, todas as regras com pesos baixos serão descartadas do modelo na fase

chamada de fase de 'Poda de Regras'. Isso assegura que um modelo com pesos baixos entre a Camada de Regras e a Camada de Saída Fuzzy irá resultar em um sistema de inferência fuzzy mais interpretável por ter um número reduzido de regras.

Dessa maneira a função de aptidão é responsável por minimizar o erro e maximizar a interpretabilidade do sistema de inferência fuzzy gerado pela RBFuzzy. Modificando os valores dos pesos α e λ os valores de acurácia e interpretabilidade podem ser ajustados para se adaptar melhor a diferentes problemas. Caso em um dado modelo a interpretabilidade seja um problema, o valor de λ pode ser aumentado para que regras mais interpretáveis sejam geradas.

Depois que o algoritmo $ACO_{\mathbb{R}}V$ achou os pesos ótimos para a rede a fase de 'Poda de Regras' ocorre. Essa é a fase final do algoritmo de aprendizagem, onde cada regra com um peso abaixo de um valor limite escolhido pelo usuário é descartada do modelo.Uma vez que a função de aptidão favorece modelos com valores mais baixos de pesos, espera-se que essa fase corte um número significativo de regras sem degradação significativa na acurácia do modelo gerado.

3.1.3 Interpretação Fuzzy

Depois do treinamento a rede pode ser interpretada como um sistema de inferência do tipo Mamdani, onde as RBFs da Camada de Regras representam as funções de pertinência da parte antecedente e os neurônios da Camada de Saída Fuzzy representam as funções da parte consequente para cada regra. A saída crisp pode ser obtida pela defuzzificação do consequente. As regras da RBFuzzy são definidas na forma:

$$\text{Regra 1} = \begin{cases} \textbf{SE } \mathrm{x_1} \text{ ESTÁ EM gaussmf}(\mathrm{C_1}, \mathrm{S_1}) \text{ E} \\ \textbf{SE } \mathrm{x_2} \text{ ESTÁ EM gaussmf}(\mathrm{C_2}, \mathrm{S_2}) \text{ E} \\ \vdots \\ \textbf{SE } \mathrm{x_k} \text{ ESTÁ EM gaussmf}(\mathrm{C_k}, \mathrm{S_k}) \\ \textbf{ENTÃO } \mathrm{y} \text{ É ALTO COM w} = (\mathrm{w_{11}}) \end{cases}$$

$$\text{Regra 2} = \begin{cases} \textbf{SE } \mathrm{x_1} \text{ ESTÁ EM gaussmf}(\mathrm{C_1}, \mathrm{S_1}) \text{ E} \\ \textbf{SE } \mathrm{x_2} \text{ ESTÁ EM gaussmf}(\mathrm{C_2}, \mathrm{S_2}) \text{ E} \\ \vdots \\ \textbf{SE } \mathrm{x_k} \text{ ESTÁ EM gaussmf}(\mathrm{C_k}, \mathrm{S_k}) \\ \textbf{ENTÃO } \text{y É MÉDIO COM w} = (\mathrm{w_{21}}) \end{cases}$$

$$\text{Regra 3} = \begin{cases} \textbf{SE } \mathrm{x_1} \text{ ESTÁ EM gaussmf}(\mathrm{C_1}, \mathrm{S_1}) \text{ E} \\ \textbf{SE } \mathrm{x_2} \text{ ESTÁ EM gaussmf}(\mathrm{C_2}, \mathrm{S_2}) \text{ E} \\ \vdots \\ \textbf{SE } \mathrm{x_k} \text{ ESTÁ EM gaussmf}(\mathrm{C_k}, \mathrm{S_k}) \\ \textbf{ENTÃO } \text{y É BAIXO COM w} = (\mathrm{w_{31}}) \end{cases}$$

x_k sendo a variável de entrada, C_k e S_k sendo o centro e largura de cada entrada respectivamente, $gaussmf$ a função de pertinência gaussiana de entrada, k o número de variáveis de entrada, y a saída e w_ij os pesos encontrados pelo algoritmo $\mathrm{ACO_{\mathbb{R}}V}$.

1: **for all** regra$_i$ na CAMADA DE REGRAS **do**
2: **for all** entrada$_i$ na CAMADA DE ENTRADA **do**
3: C_i = Centro da distribuição do cluster$_i$
4: S_i = Desvio Padrão da distribuição do cluster$_i$
5: Antecedente$_i$ = gaussmf(C_i, S_i)
6: **end for**
7: **for all** saida$_i$ na CAMADA DE SAÍDA FUZZY **do**
8: Defina o peso da função de pertinência de saída igual a conexão do peso entre regra$_i$ e saida$_i$
9: **end for**
10: Escrever a regra gerada
11: **end for**

Figura 12: RBFuzzy Algoritmo de extração de regras
Fonte: Elaborada pelo autor

Um exemplo mais prático é dado a seguir:

$$\begin{cases} \textbf{SE } \text{a temperatura do motor ESTÁ EM gaussmf}(90^\mathrm{o}\mathrm{C}, 5^\mathrm{o}\mathrm{C}) \\ \text{E } \textbf{SE } \text{a pressão do motor ESTÁ EM gaussmf}(130\mathrm{psi}, 8\mathrm{psi}) \\ \textbf{ENTÃO } \text{o fluxo de gasolina É BAIXO COM w} = 0.89 \end{cases}$$

O algoritmo para converter uma rede treinada em sua interpretação fuzzy é demostrado na Figura 12.

3.1.4 Vantagens e Desvantagens

Cada modelo possui suas vantagens e desvantagens que devem ser consideradas. Quando a RBFuzzy é comparada com outras redes Neuro-Fuzzy encontradas na literatura, é possível citar algumas vantagens e desvantagens.

Vantagens:

- Algoritmo de aprendizado multi-objetivo: É possível adicionar mais objetivos para o algoritmo de aprendizagem na função de aptidão. Esse objetivo pode ser específico pelo problema a ser resolvido pela RBFuzzy.
- Regras altamente interpretáveis: As regras tendem a ter alta interpretabilidade pelo fato de que um dos objetivos padrão do algoritmo de aprendizagem é melhorar a interpretabilidade.
- Facilidade de implementação: A RBFuzzy tem uma arquitetura simples, porém sólida, o que simplifica a implementação do modelo.

Desvantagens:

- Esquece conjuntos de treinamentos passados: Se novos dados forem apresentados para o treinamento, os conjuntos de dados anteriores são esquecidos. Quando um conjunto de dados novo é apresentado, os dados antigos devem ser reapresentados para que não ocorra o 'esquecimento'.
- Sem treinamento *Online*: A RBFuzzy não pode ser utilizada em sistemas onde dados são continuamente gerados e apresentados para a rede, tais como sistemas de tempo-real. A RBFuzzy necessita de uma fase específica para treinamento.

3.2 CONCLUSÃO

Neste capítulo foi apresentada a proposta de uma nova rede Neuro-Fuzzy, chamada RBFuzzy. Sua arquitetura foi definida, assim como os algoritmos necessários para a construção e treinamento da rede.

Ao final do capítulo vantagens e desvantagens foram apresentados para reforçar os pontos fortes desta nova rede Neuro-Fuzzy.

No próximo capítulo serão apresentados experimentos e resultados obtidos com a RBFuzzy para demonstrar, utilizando dados de problemas clássicos e dados de problemas reais, alguns tipos de problemas onde a RBFuzzy pode ser utilizada como solução.

4 EXPERIMENTOS E RESULTADOS

Para testar a performance da RBFuzzy, experimentos utilizando dados simulados e reais foram feitos. Os experimentos serão mostrados nas próximas seções.

São apresentados experimentos usando dados simulados do problema da Gorjeta (GULLEY; JANG; WANG, 1998), da série temporal Mackey-Glass (MACKEY; GLASS et al., 1977) e do Engine dataset (BEALE; HAGAN; DEMUTH, 1992), onde a rede RBFuzzy foi testada contra os resultados da rede EFuNN. Um experimento usando dados reais de um conjunto de dados de perfuração de petróleo também foram feitos.

4.1 PROBLEMA DA GORJETA

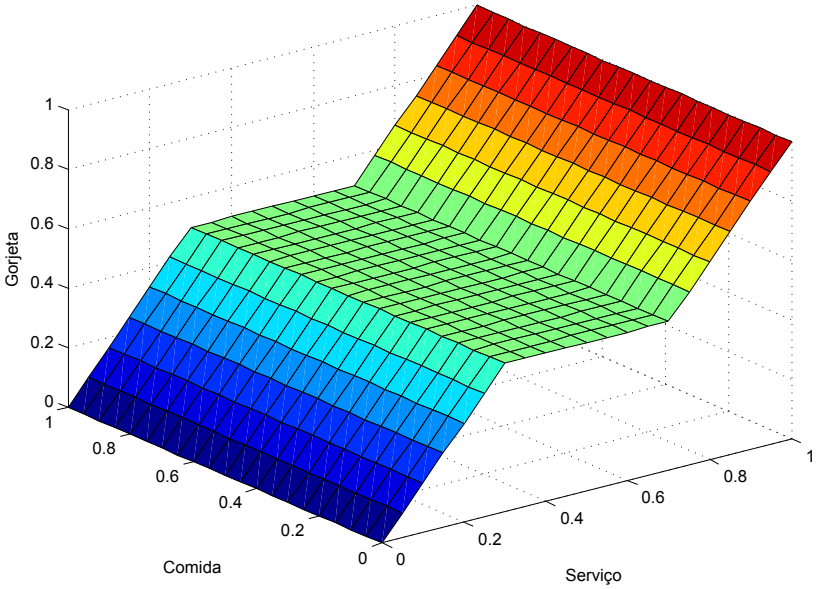

Figura 13: Superfície esperada do problema da Gorjeta.
Fonte: Elaborada pelo autor

Nesta seção é apresentada a resolução do problema da Gorjeta pela rede RBFuzzy. O problema da Gorjeta é descrito em (GULLEY;

JANG; WANG, 1998). O problema consiste em decidir o melhor valor de gorjeta a ser dado para o garçom levando em consideração a qualidade do serviço e a qualidade da comida. A Figura 13 mostra qual é a superfície de resposta da resolução do problema de acordo com a entrada da qualidade do serviço e comida.

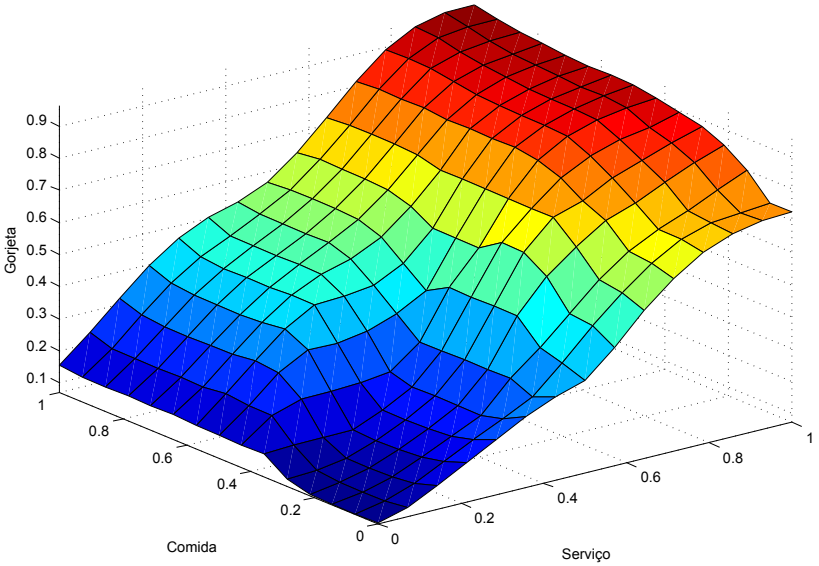

Figura 14: Superfície gerada pela RBFuzzy para a resolução do problema da gorjeta utilizando 6 regras.
Fonte: Elaborada pelo autor

Para resolver esse problema, foi utilizado como conjunto de treinamento 200 pontos escolhidos de maneira aleatória. Os parâmetros utilizados na RBFuzzy foram os pesos da função de aptidão de 0.8 para melhorar a acurácia e 0.2 para melhorar a interpretabilidade e 15 neurônios na camada de regras. Os valores dos pesos da função de aptidão para acurácia e para interpretabilidade, assim como o número de neurônios na camada de regras, foram encontrados através de experimentação.

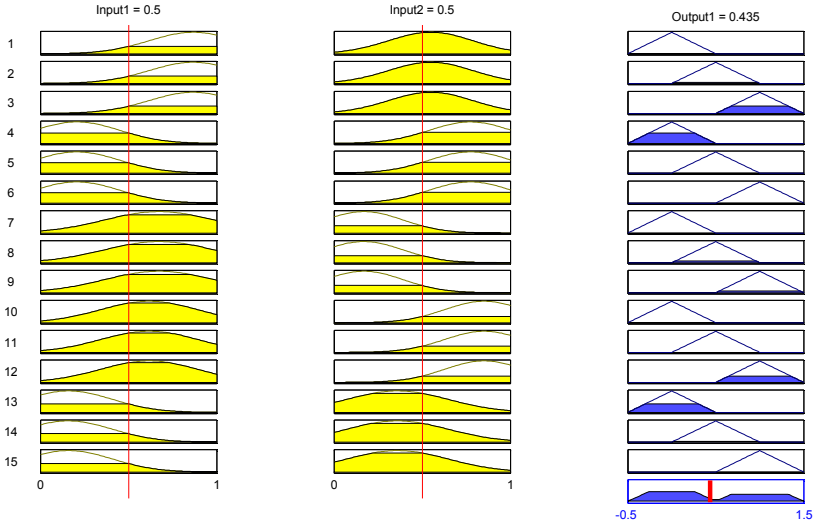

Figura 15: Regras geradas pela RBFuzzy para a resolução do problema da gorjeta antes do passo de Poda de Regras

Fonte: Elaborada pelo autor

Como resultado foram geradas 6 regras fuzzy que podem ser vistas na Figura 16 e a superfície de resposta que pode ser vista na Figura 14. Como comparativo na Figura 15 são mostradas as regras que haviam sido geradas antes do passo de Poda de Regras, com um total de 15 regras. O erro médio quadrático antes do passo de Poda de Regras era de 0.00294 e o resultado final do erro médio quadrático após o passo de Poda de Regras foi de 0.0030. Isso demonstra que a fase de Poda de Regras removeu uma quantidade significativa de regras (reduziu em 60% o número de regras) ao mesmo tempo que não interferiu de maneira significativa na acurácia da rede (houve um aumento de 2% no erro médio quadrático).

As seis regras exibidas graficamente na Figura 16 possuem a seguinte representação textual:

1. Se **Comida** e gaussmf(0.8678, 0.2511) e **Serviço** é gaussmf(0.5476, 0.2392) então **Gorjeta** é Média com w = 0.13
2. Se **Comida** é gaussmf(0.2065, 0.2447) e **Serviço** é gaussmf(0.7723, 0.2344) então **Gorjeta** é Baixa com w = 1
3. Se **Comida** é gaussmf(0.6737, 0.3113) e **Serviço** é gaussmf(0.1678, 0.2245) então **Gorjeta** é Média com w = 0.27

4. Se **Comida** é gaussmf(0.6737, 0.3113) e **Serviço** é gaussmf(0.1678, 0.2245) então **Gorjeta** é Alta com w = 0.25

5. Se **Comida** é gaussmf(0.6169, 0.2754) e **Serviço** é gaussmf(0.8515, 0.2250) então **Gorjeta** é Alta com w = 1

6. Se **Comida** é gaussmf(0.1591, 0.2500) e **Serviço** é gaussmf(0.3592, 0.2868) então **Gorjeta** é Baixa com w = 1

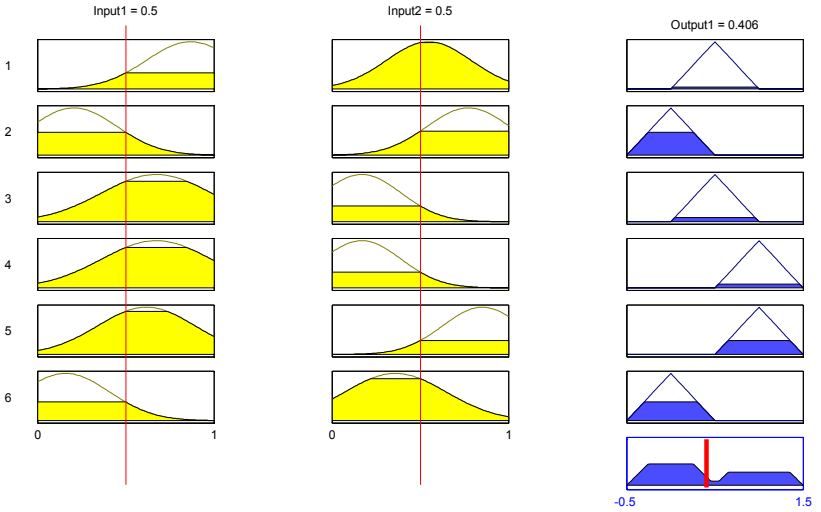

Figura 16: Regras geradas pela RBFuzzy para a resolução do problema da gorjeta após o passo de Poda de Regras.
Fonte: Elaborada pelo autor

4.2 PREVISÃO DA SÉRIE TEMPORAL MACKEY-GLASS

Nesta seção é avaliada a performance da rede RBFuzzy utilizando a série temporal Mackey-Glass (MG)(MACKEY; GLASS et al., 1977). Essa série é definida pela equação diferencial 4.1:

$$\frac{\partial x}{\partial t} = \beta \frac{x_\tau}{1 + x_\tau^n} - \gamma x(t), \quad \gamma, \beta, n > 0, \tag{4.1}$$

onde β, γ, τ, n são números reais e x_τ representa o valor da variáve x no tempo $(t - \tau)$. Os experimentos foram conduzidos utilizando um

conjunto de dados com 1000 dados. O mesmo conjunto de dados foi utilizado para a avaliação de ambos modelos.

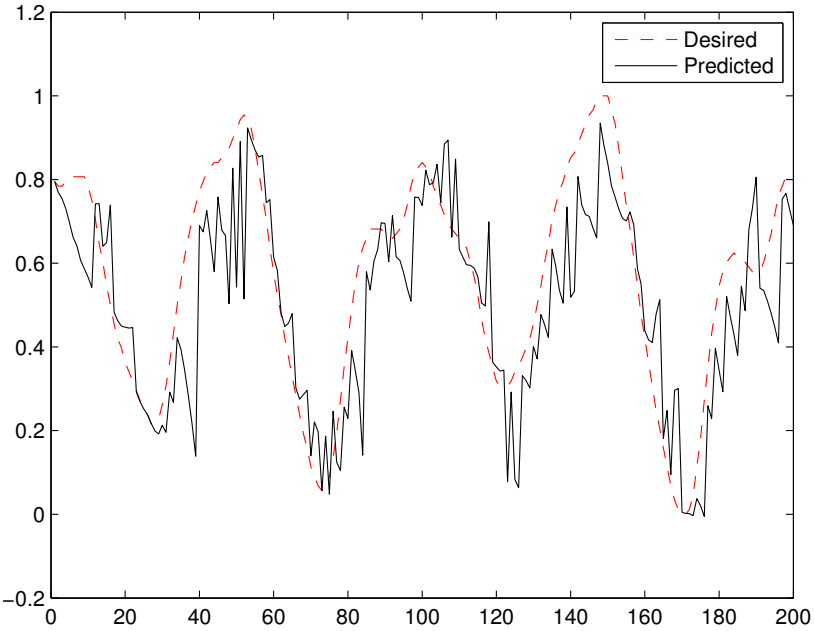

Figura 17: Previsão da EFuNN para os dados da série Mackey-Glass
Fonte: Elaborada pelo autor

A qualidade da previsão foi avaliada utilizando o índice de erro não-dimensional (NDEI). NDEI é definido como sendo o erro médio quadrático dividido pelo desvio padrão da série alvo, como mostrado na equação 4.2:

$$NDEI = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(o_i - t_i)^2}}{\sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}}} \tag{4.2}$$

onde N é o número total de pontos de treinamento contidos no conjunto de dados de treinamento, o_i é o valor de saída para o dado de treinamento i, t_i é o valor esperado para a saída do dado de treinamento i, x_i é o

valor da saída do dado de treinamento i e $\bar{x}$ é a média do conjunto de dados de treinamento.

Este índice foi adotado pois uma série de artigos que também abordam redes Neuro-Fuzzy o utilizam (WATTS, 2009).

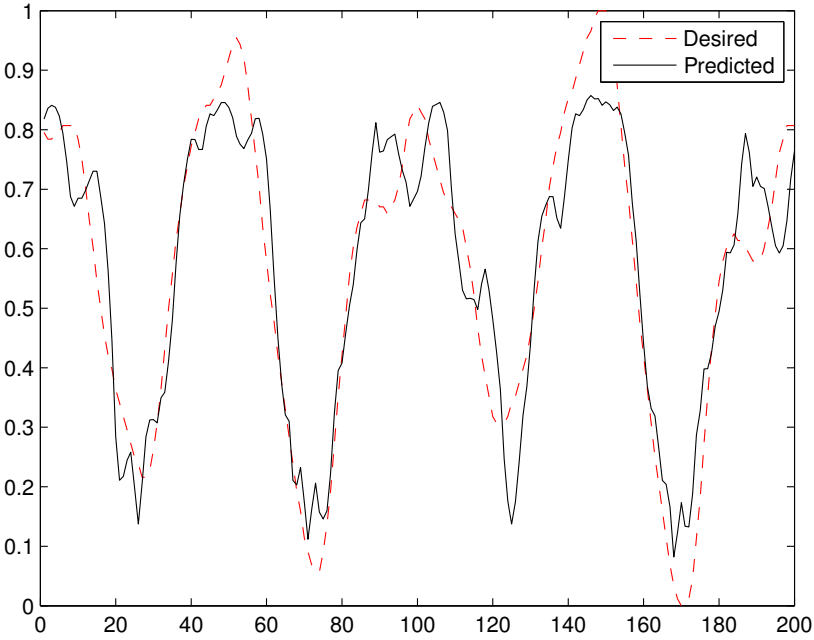

Figura 18: Previsão da RBFUZZY para os dados da série Mackey-Glass
Fonte: Elaborada pelo autor

Rede	# de Regras	NDEI
RBFuzzy	7	0.3684
EFuNN	53	0.4481

Tabela 1: Resultados dos testes com os dados da série Mackey-Glass

Os resultados dos experimentos estão resumidos na Tabela 1 e as Figuras 17 e 18 mostram os resultados para a EFuNN e RBFuzzy respectivamente. Analisando os resultados gráficos pode-se concluir que ambos modelos ofereceram aproximações boas. A RBFuzzy teve problemas ao prever os extremos da série. Acredita-se que a causa deste comportamento são os valores encontrados pelo algoritmo de

clusterização para os valores de centro e spread das funções de pertinência de entrada.

A EFuNN atingiu um NDEI de 0.4481, porém falhou em dar um número bom de regras, gerando no total 53 regras fuzzy enquanto a RBFuzzy gerou apenas 7 regras e atingiu um NDEI de 0.3684. A RBFuzzy obteve melhor interpretabilidade devido ao número reduzido de regras quando comparada com a EFuNN. A função de aptidão desse experimento tinha um peso de 0.8 para melhorar a acurácia e 0.2 para melhorar a intepretabilidade.

A Figura 20 mostra as regras geradas pela RBFuzzy usando a Toolbox Fuzzy do Matlab. Para questões de comparação, a Figura 19 mostra as regras geradas pela RBFuzzy antes da fase final do algoritmo de aprendizagem, onde o corte de regras é aplicado. Um conjunto com 36 regras é reduzido a um conjunto de 7 regras nessa fase final. O valor utilizado como limite definido para o descarte de regras foi de 0.1 e após o descarte das regras com pesos abaixo desse valor a diferença no resultado entre o modelo com 36 regras e o modelo com 7 regras foi de apenas 1.2%. Isso demonstra que a função de aptidão teve bom sucesso ao melhorar a interpretabilidade do sistema de inferência fuzzy resultante sem comprometer a acurácia.

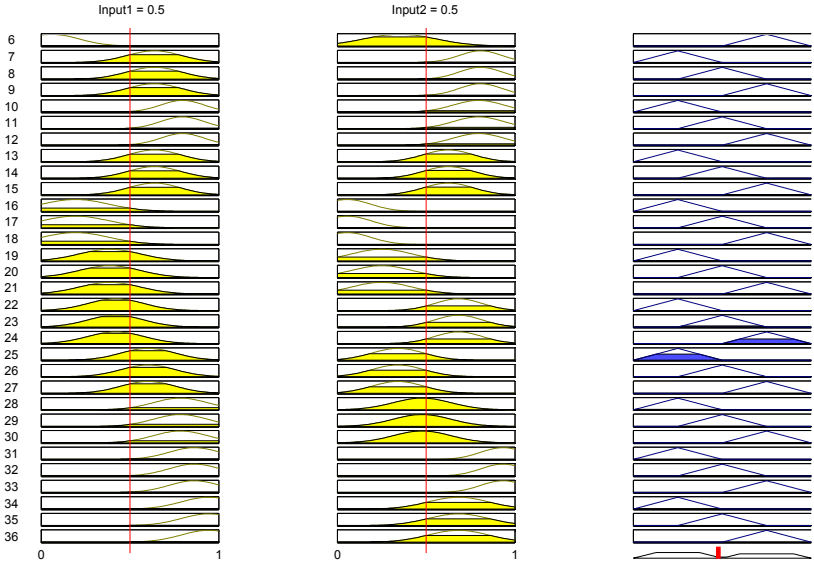

Figura 19: Conjunto de regras da rede RBFuzzy para a série Mackey-Glass antes da fase de 'Poda de Regras'

Fonte: Elaborada pelo autor

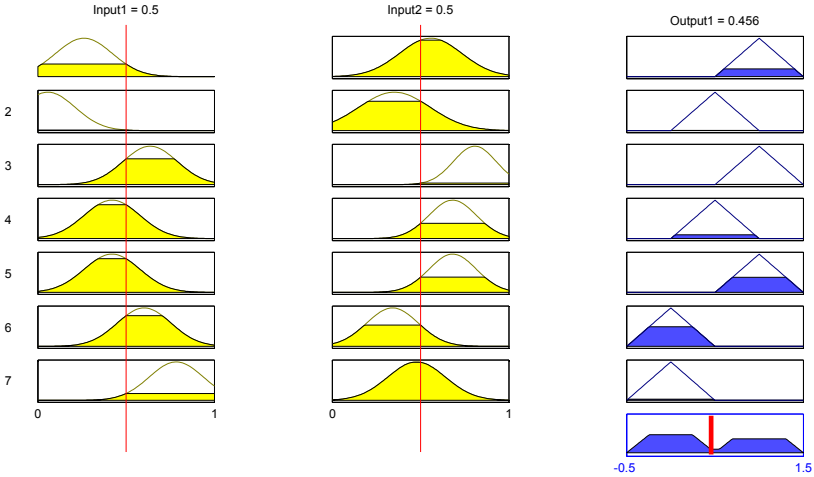

Figura 20: Conjunto de regras da rede RBFuzzy para a série Mackey-Glass depois da fase de 'Poda de Regras'

Fonte: Elaborada pelo autor

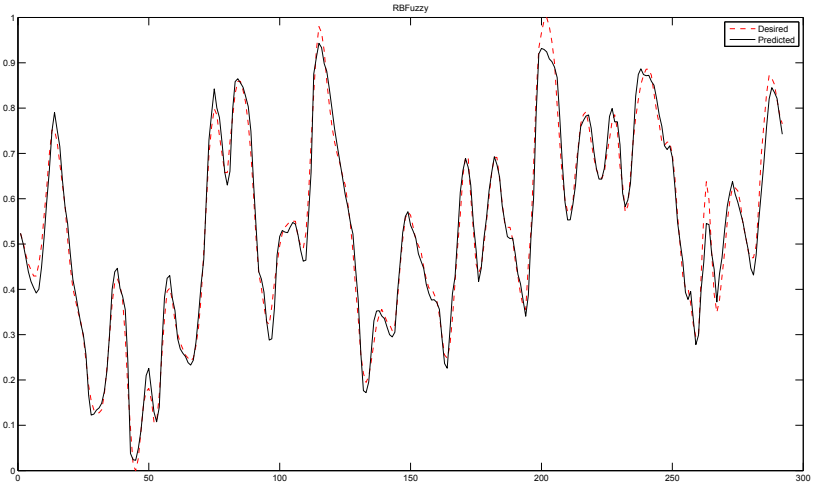

Figura 21: Previsão da rede RBFUZZY para o Engine Dataset
Fonte: Elaborada pelo autor

4.3 PREVISÃO DO ENGINE DATASET

O experimento conduzido nesta seção utilizou o Engine Dataset (BEALE; HAGAN; DEMUTH, 1992) com um número de 1199 dados, tendo como entrada a taxa de combustível para prever as emissões de óxido nitroso.

A rede RBFuzzy gerou 6 regras fuzzy e atingiu um NDEI de 0.4225. A função de aptidão utilizada neste experimento utilizou um valor de 0.8 como peso para acurácia e 0.2 como peso para interpretabilidade.

O resultado gráfico pode ser visto na Figura 21, que demonstra que a rede RBFuzzy dá uma boa aproximação para os dados utilizando apenas 6 regras fuzzy.

4.4 USO NA PERFURAÇÃO DE POÇOS DE PETRÓLEO

Para reduzir custos na perfuração *offshore* de poços de petróleo o tempo requerido para perfurar com sucesso um determinado poço deve ser estimado de maneira precisa, uma vez que a maior parte dos custos associados a esta operação estão ligados ao valor de aluguel dos equipamentos necessários para a perfuração (GANDELMAN, 2012).

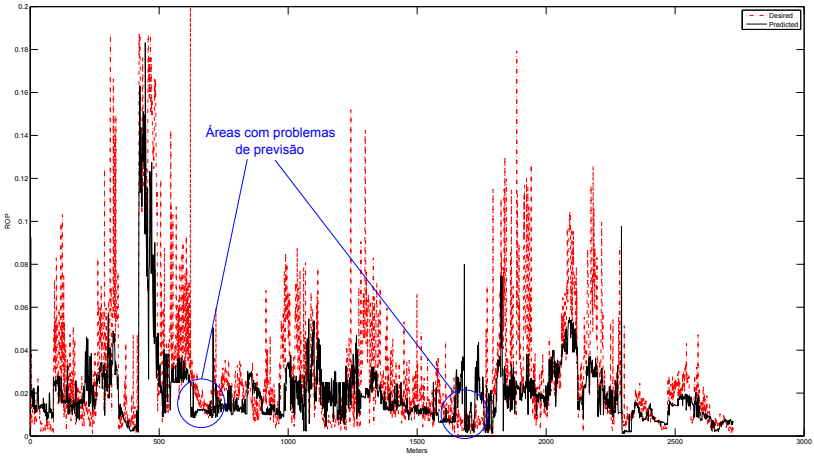

Figura 22: Previsão da RBFUZZY para o conjunto de treinamento do valor de ROP
Fonte: Elaborada pelo autor

Entretanto, cada perfuração possui propriedades únicas que fazem com que essa tarefa seja extremamente difícil. Muitas propriedades variam, tais como tipo de rocha, porosidade da rocha, presença de gás, pressão, taxa de desgaste da broca entre outros. Todas essas propriedades afetam a taxa de penetração (ROP), que é a velocidade com que a broca quebra a rocha com o objetivo de aumentar a profundidade da perfuração (Bourgoyne Jr.; Young Jr., 1974). Há outros parâmetros que também afetam a ROP e que podem ser controlados por um operador de perfuração (BILGESU et al., 1997): peso sobre a broca (PSB), revoluções por minuto (RPM), tipo da broca, diâmetro da broca e pressão do fluído de perfuração.

A maior parte do trabalho do planejamento de uma perfuração está restrita a ajustar o tipo de broca, RPM e PSB para atingir um valor aceitável de ROP. Para otimizar esse trabalho muitos sistemas utilizando redes neurais artificiais foram propostos na literatura (BILGESU et al., 1997) e até escolhiam parâmetros automaticamente de RPM e PSB (FONSECA et al., 2006).

A RBFuzzy foi testada utilizando os dados reais disponibilizados pela empresa Petrobras tendo como resultado a previsão mostrada na figura 22, cujo modelo resultou em 35 regras. Os dados que foram

disponibilizados são provenientes da camada de pré-sal, onde as profundidades de perfuração são maiores que 5000 metros, cruzando camadas de sal que podem chegar a 2000 metros (BELTRAO et al., 2009). Devido a essas condições extremas, a qualidade ruim dos dados é um problema inerente e que não pode ser evitado.

A RBFuzzy é utilizada para superar esse problema disponibilizando uma interpretação linguística para o modelo de previsão para o especialista em perfuração de poços. O especialista pode ler o modelo e corrigir os erros gerados pelas partes ruins/ruidosas dos dados. Se alguma variável de entrada está faltando dos dados o especialista pode inserí-la no modelo utilizando seu conhecimento explícito do problema. Depois que o modelo é corrigido pelo especialista ele pode ser utilizado para fazer previsões mais precisas.

No resultado mostrado na figura 22, duas áreas com problemas de previsão foram destacadas. Utilizando o modelo gerado, um especialista tem a possibilidade de verificar os dados que geraram as regras nas áreas destacadas e as regras que foram geradas afim de corrigir essa falha na previsão. O problema pode ser, por exemplo, proveniente de erros nos dados ou falta de dados de entrada importantes para explicar o comportamento da perfuração nessa área.

5 CONSIDERAÇÕES FINAIS

Esse trabalho apresentou um novo modelo de rede Neuro-Fuzzy que utiliza o sistema de inferência fuzzy do tipo Mamdani. Empregando funções de base radial como ativação dos neurônios da camada intermediária e um algoritmo de otimização no treinamento da rede, foi mostrado que é possível extrair conhecimento dos dados na forma de regras com um maior nível de acurácia e interpretabilidade quando comparado com outros métodos da literatura.

Para definir os parâmetros das funções de ativação dos neurônios da camada intermediária um algoritmo de clusterização foi utilizado. De acordo com os agrupamentos encontrados nos dados de entrada são definidos os centros e aberturas das funções de base radial.

Para o ajuste dos pesos das conexões neuronais da camada de saída, foi utilizado o algoritmo de otimização chamado $\text{ACO}_{\mathbb{R}}\text{V}$. Esse algoritmo é uma extensão do algoritmo da otimização da colônia de formigas e seu uso foi feito para incorporar mais de um objetivo no algoritmo de aprendizado, otimizando dessa forma a acurácia e a interpretabilidade ao mesmo tempo.

Esse método possui muitas vantagens, por ser um tipo de treinamento onde se pode buscar mais de um objetivo (multi-objetivo), é possível colocar novas condições na otimização para melhorar a extração de conhecimento. Isso pode ser feito adicionando novos objetivos na função de aptidão do algoritmo.

Os resultados dos experimentos demonstraram que o nível de acurácia e interpretabilidade atingidos foram melhores quando comparados à rede EFuNN.

Algumas limitações do modelo proposto merecem ser citadas como proposta para trabalhos futuros:

- As funções de pertinência de entrada aumentam de acordo com o número de regras do modelo. A aplicação de algum método para diminuição do número dessas funções de pertinência aumentaria ainda mais a interpretabilidade.

- O número de regras deve ser escolhido pelo usuário do modelo. Alguns algoritmos, como por exemplo o algoritmo de clusterização ECM, talvez possam achar automaticamente o número de regras mais adequado de acordo com os dados do problema a ser resolvido.

- Nas funções de pertinência de entrada são utilizadas apenas funções gaussianas e nas funções de pertinência de saída apenas funções

triangulares. Tipos diferentes de funções de pertinência podem apresentar resultados melhores dependendo do problema a ser resolvido.

- O modelo suporta apenas o tipo de aprendizado *Offline*. O tipo de aprendizado *Online* possui muitas vantagens e poderia ser utilizado em conjunto com o aprendizado *Offline*.

REFERÊNCIAS

ALMOMANI, A. et al. Evolving fuzzy neural network for phishing emails detection. **Journal of Computer Science**, v. 8, n. 7, p. 1099, 2012.

ALONSO, J. M.; MAGDALENA, L.; GONZÁLEZ-RODRÍGUEZ, G. Looking for a good fuzzy system interpretability index: An experimental approach. **International Journal of Approximate Reasoning**, Elsevier, v. 51, n. 1, p. 115–134, 2009.

ARTHUR, D.; VASSILVITSKII, S. k-means++: The advantages of careful seeding. In: SOCIETY FOR INDUSTRIAL AND APPLIED MATHEMATICS. **Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms**. [S.l.], 2007. p. 1027–1035.

BEALE, M.; HAGAN, M. T.; DEMUTH, H. B. Neural network toolbox. **Neural Network Toolbox, The Math Works**, p. 5–25, 1992.

BELTRAO, R. L. et al. Ss: Pre-salt santos basin-challenges and new technologies for the development of the pre-salt cluster, santos basin, brazil. In: **Offshore Technology Conference**. [S.l.: s.n.], 2009.

BERTHOLD, M. R.; HUBER, K.-P. Constructing fuzzy graphs from examples. **Intelligent Data Analysis**, IOS Press, v. 3, n. 1, p. 37–53, 1999.

BILGESU, H. et al. A New Approach for the Prediction of Rate of Penetration (ROP) Values. In: **SPE Eastern Regional Meeting**. Lexington, Kentucky: Society of Petroleum Engineers, 1997. ISBN 9781555633981.

Bourgoyne Jr., A.; Young Jr., F. A Multiple Regression Approach to Optimal Drilling and Abnormal Pressure Detection. **SPE Journal**, v. 14, n. 4, p. 371–384, 1974.

CASILLAS, J. et al. **Interpretability improvements to find the balance interpretability-accuracy in fuzzy modeling: an overview**. [S.l.]: Springer, 2003.

CASTRO, J. L.; DELGADO, M. Fuzzy systems with defuzzification are universal approximators. **Systems, Man, and Cybernetics,**

Part B: Cybernetics, IEEE Transactions on, IEEE, v. 26, n. 1, p. 149–152, 1996.

CASTRO, J. L.; MANTAS, C. J.; BENÍTEZ, J. M. Interpretation of artificial neural networks by means of fuzzy rules. **Neural Networks, IEEE Transactions on**, IEEE, v. 13, n. 1, p. 101–116, 2002.

CHANG, F.-J.; WANG, K.-W. A systematical water allocation scheme for drought mitigation. **Journal of Hydrology**, v. 507, n. 0, p. 124 – 133, 2013. ISSN 0022-1694.

CHANG, P.-C.; FAN, C.-Y.; LIN, J.-J. Monthly electricity demand forecasting based on a weighted evolving fuzzy neural network approach. **International Journal of Electrical Power & Energy Systems**, Elsevier, v. 33, n. 1, p. 17–27, 2011.

CONTI, C. R.; ROISENBERG, M.; NETO, G. S. Aco$_{\mathbb{R}}$v-an algorithm that incorporates the visibility heuristic to the aco in continuous domain. In: IEEE. **Evolutionary Computation (CEC), 2012 IEEE Congress on**. [S.l.], 2012. p. 1–8.

DORIGO, M. Optimization, learning and natural algorithms. **Ph. D. Thesis, Politecnico di Milano, Italy**, 1992.

DORIGO, M.; GAMBARDELLA, L. M. Ant colonies for the travelling salesman problem. **BioSystems**, Elsevier, v. 43, n. 2, p. 73–81, 1997.

FONSECA, T. C. et al. A Genetic Neuro-Model Reference Adaptive Controller for Petroleum Wells Drilling Operations. In: **International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce**. [S.l.: s.n.], 2006. p. 3. ISBN 0769527310.

GACTO, M. J.; ALCALÁ, R.; HERRERA, F. Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures. **Information Sciences**, Elsevier, v. 181, n. 20, p. 4340–4360, 2011.

GANDELMAN, R. A. **Predição da ROP e otimização em tempo real de parâmetros operacionais na perfuração de poços de petróleo offshore**. 195 p. Tese (Doutorado) — UFRJ, 2012.

GEHRKE, M.; WALKER, C.; WALKER, E. Algebraic aspects of fuzzy sets and fuzzy logics. In: CITESEER. **Proceedings of the**

Workshop on Current Trends and Developments in Fuzzy Logic, Thessaloniki, Greece, October. [S.l.], 1999. v. 16, n. 20, p. 1998.

GUILLAUME, S. Designing fuzzy inference systems from data: an interpretability-oriented review. **Fuzzy Systems, IEEE Transactions on**, IEEE, v. 9, n. 3, p. 426–443, 2001.

GUILLAUME, S.; MAGDALENA, L. Expert guided integration of induced knowledge into a fuzzy knowledge base. **Soft computing**, Springer, v. 10, n. 9, p. 773–784, 2006.

GULLEY, N.; JANG, J.-S. R.; WANG, W.-c. **Fuzzy Logic Toolbox: for Use with MATLAB: User's Guide**. [S.l.]: MathWorks, Incorporated, 1998.

JANG, J.-S.; SUN, C.-T. Neuro-fuzzy modeling and control. **Proceedings of the IEEE**, IEEE, v. 83, n. 3, p. 378–406, 1995.

JANG, J.-S. R. Anfis: adaptive-network-based fuzzy inference system. **IEEE Transactions on Systems, Man and Cybernetics**, v. 23, n. 3, p. 665–685, 1993.

JASSBI, J. et al. Transformation of a mamdani fis to first order sugeno fis. In: IEEE. **Fuzzy Systems Conference, 2007. FUZZ-IEEE 2007. IEEE International**. [S.l.], 2007. p. 1–6.

JASSBI, J. et al. A comparison of mandani and sugeno inference systems for a space fault detection application. In: IEEE. **Automation Congress, 2006. WAC'06. World**. [S.l.], 2006. p. 1–8.

KASABOV, N.; WOODFORD, B. Rule insertion and rule extraction from evolving fuzzy neural networks: algorithms and applications for building adaptive, intelligent expert systems. In: IEEE. **Fuzzy Systems Conference Proceedings, 1999. FUZZ-IEEE'99. 1999 IEEE International**. [S.l.], 1999. v. 3, p. 1406–1411.

KASABOV, N. K. The ECOS Framework and the ECO Learning Method for Evolving Connectionist Systems. **Journal of Advanced Computational Intelligence and Intelligent Informatics**, v. 2, p. 195–202, 1998.

KASABOV, N. K. **Evolving Connectionist Systems: The Knowledge Engineering Approach**. [S.l.]: Springer London, 2007. ISBN 978-1-84628-345-1.

KASABOV, N. K.; SONG, Q. Denfis: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. **IEEE Transactions on Fuzzy Systems**, v. 10, n. 2, p. 144–154, 2002.

KOHAVI, R.; PROVOST, F. Glossary of terms. **Machine Learning**, v. 30, n. 2-3, p. 271–274, 1998.

LEEKWIJCK, W. V.; KERRE, E. E. Defuzzification: criteria and classification. **Fuzzy sets and systems**, Elsevier, v. 108, n. 2, p. 159–178, 1999.

ŁESKI, J. ε–insensitive learning techniques for approximate reasoning systems. **Int. J. Comput. Cognit**, v. 1, n. 1, p. 21–77, 2003.

LI, W.; HORI, Y. An algorithm for extracting fuzzy rules based on rbf neural network. **Industrial Electronics, IEEE Transactions on**, IEEE, v. 53, n. 4, p. 1269–1276, 2006.

MACKEY, M. C.; GLASS, L. et al. Oscillation and chaos in physiological control systems. **Science**, v. 197, n. 4300, p. 287–289, 1977.

MAMDANI, E. H. Application of fuzzy algorithms for control of simple dynamic plant. **Electrical Engineers, Proceedings of the Institution of**, IET, v. 121, n. 12, p. 1585–1588, 1974.

MAMDANI, E. H. Application of fuzzy logic to approximate reasoning using linguistic synthesis. **Computers, IEEE Transactions on**, IEEE, v. 100, n. 12, p. 1182–1191, 1977.

MIKUT, R.; JÄKEL, J.; GRÖLL, L. Interpretability issues in data-based learning of fuzzy systems. **Fuzzy Sets and Systems**, Elsevier, v. 150, n. 2, p. 179–197, 2005.

MITCHELL, T. M. **The Discipline of Machine Learning. July 2006**. [S.l.], 2009.

OCAK, H.; ERTUNC, H. M. Prediction of fetal state from the cardiotocogram recordings using adaptive neuro-fuzzy inference systems. **Neural Computing and Applications**, Springer, p. 1–7, 2012.

PETROVIC-LAZAREVIC, S.; ZHANG, J. Y. Neuro-fuzzy models and tobacco control. In: SPRINGER. **Proceedings of the European Computing Conference**. [S.l.], 2009. p. 25–31.

SUGENO, M.; TAKAGI, T. A new approach to design of fuzzy controller. In: **Advances in Fuzzy Sets, Possibility Theory, and Applications**. [S.l.]: Springer, 1983. p. 325–334.

TANSCHEIT, R. Sistemas fuzzy. **Inteligência computacional: aplicada à administração, economia e engenharia em Matlab**, p. 229–264, 2004.

TSOUKALAS, L. H.; UHRIG, R. E. **Fuzzy and neural approaches in engineering**. [S.l.]: John Wiley & Sons, Inc., 1996.

TSUKAMOTO, Y. An approach to fuzzy reasoning method. **Advances in fuzzy set theory and applications**, v. 137, p. 149, 1979.

WATTS, M. J. A decade of kasabov's evolving connectionist systems: A review. **IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, v. 39, n. 3, p. 253–269, 2009.

ZADEH, L. A. Fuzzy sets. **Information and Control**, v. 8, n. 3, p. 338–353, 1965.

ZADEH, L. A. Outline of a new approach to the analysis of complex systems and decision processes. **Systems, Man and Cybernetics, IEEE Transactions on**, IEEE, n. 1, p. 28–44, 1973.

ZADEH, L. A. The concept of a linguistic variable and its application to approximate reasoning - i. **Information sciences**, Elsevier, v. 8, n. 3, p. 199–249, 1975.

ZHOU, S.-M.; GAN, J. Q. Low-level interpretability and high-level interpretability: a unified view of data-driven interpretable fuzzy system modelling. **Fuzzy Sets and Systems**, Elsevier, v. 159, n. 23, p. 3091–3131, 2008.

ZUO, Q. Description of strictly monotonic interval and/or operations. **Reliable Computing**, p. 23–25, 1995.